

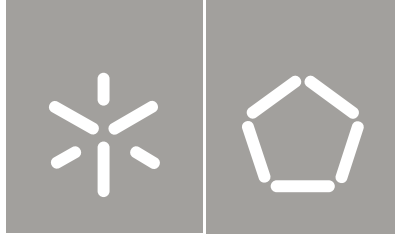


Universidade do Minho
Escola de Engenharia

Luís Gonzaga Martins Ferreira
Architectures for Integration of Information Systems under
conditions of Dynamic Reconfiguration of Virtual Enterprises

Luís Gonzaga Martins Ferreira

Architectures for Integration of Information
Systems under conditions of Dynamic
Reconfiguration of Virtual Enterprises



Universidade do Minho
Escola de Engenharia

Luís Gonzaga Martins Ferreira

Architectures for Integration of Information
Systems under conditions of Dynamic
Reconfiguration of Virtual Enterprises

Thesis submitted in fulfilment of the requirements for the degree
of Doctorate in Philosophy in
Doctoral Programme in Industrial and Systems Engineering

Supervised by:
Professor Goran D. Putnik
Professor Maria Manuela Cruz Cunha

December, 2012

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA TESE/TRABALHO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE.

Universidade do Minho, 19/12/2012

Assinatura: _____

Esta tese é dedicada aos meus pais, *Helena Pinheiro* (1929-2005) e *Joaquim Ferreira* (1928) cujas vidas me trouxeram até aqui e me levarão bem mais além, e ao meu irmão *Serafim* (1961-1977) que não teve a felicidade de chegar tão longe.

This thesis is dedicated to my parents, *Helena Pinheiro* (1929-2005) and *Joaquim Ferreira* (1928) whose lives have brought me here and will lead me beyond, and to my brother *Serafim* (1961-1977) who has not had the happiness of reaching so far.

This page was intentionally left blank.

Aknowlegments

After completing this long period of research and dedication, I would like to thank several colleagues, friends and my family for their unfailing support.

In particular, *Claudio* for his very interesting points of view in the world of *User Interfaces* and *User Experience*; *Helio* for his motivation and constant alertness; friends *António Tavares* and *Joaquim Silva* for the many clashes of ideas and opinions; *Miguel Nunes* for his indispensable assistance in carrying out the experiments; *Sofia Coelho* for her cheerfulness, friendship and motivation; *Manuela Cunha* for her friendship, straightforwardness, foresight and objectivity; *Professor Goran Putnik* for his prowess and elegant wisdom; *Professor João Carvalho* for his friendship and motivation.

To IPCA (the Polytechnics Institute I work for) and the PROTEC financial programme, who together have created the necessary conditions (financial and professional) to carry out this work.

To my dedicated Masters students who made themselves available to carry out the experiments.

To my family for the support they have always expressed, especially to my twin sister who, though physically distant, made sure to always be present.

To *Romeu* for his friendship, continuous strength and motivation.

To *Ana* for her patience and great encouragement, and to our daughters *Ritinha* and *Aninhas* who will finally be able to see their father more relaxed.

Many thanks to all,

lufer

This page was intentionally left blank.

Agradecimentos

Uma vez concluído este longo período de investigação e dedicação gostaria de enaltecer o apoio conseguido por vários colegas, amigos e familiares.

Em especial ao Cláudio pelos seus interessantíssimos pontos de vista no mundo dos *User Interfaces* e *User Experience*, ao Hélio pela motivação e constante alerta; aos amigos António Tavares e Joaquim Silva pelas muitas confrontações de ideias e opiniões; ao Miguel Nunes pela sua ajuda essencial na realização das experimentações; à Sofia Coelho pela sua alegria, amizade e motivação; à Manuela Cunha pela sua simplicidade, clarividência e objectividade; ao Professor Goran Putnik pela sua capacidade e elegante sabedoria; ao Professor João Carvalho pela sua amizade e motivação.

Ao IPCA (Instituto Politécnico para o qual me orgulho trabalhar) e ao programa PROTEC, que em conjunto criaram as condições necessárias (profissionais e financeiras) para a realização deste trabalho.

Aos meus alunos de Mestrado que se disponibilizaram com afinho e dedicação à realização das experimentações.

Aos meus familiares pelo apoio que sempre manifestaram, em especial à minha irmã gémea que mesmo distante fisicamente fez questão de estar sempre presente.

Ao Romeu pela amizade e insistente força e motivação.

À minha Ana pela sua paciência e essencial encorajamento, às minhas filhas Ritinha e Aninhas que finalmente poderão ver o Pai mais descansado.

A todos, o meu muito obrigado

lufer

This page was intentionally left blank.

Abstract

The aim of this thesis is to explore Architectures of information systems Integration under conditions of dynamic reconfiguration of Virtual Enterprises. The main challenge that we identify and which formed the basis of the research is that information technologies alone cannot support efficiently and effectively the human knowledge and their natural way of interacting.

Already from Sausurre (1916) it could be argued that part of knowledge resides in person, and the attempt to try to model it is sufficient for it to be misrepresented. And this is the motto of all this work. Enhance the capabilities of emerging technologies, but in the sense that allow human-to-human interaction, having the information system merely a means to make this possible.

Thus we argue that a communicational architecture of information systems integration (where Pragmatics mechanisms are enabled) in virtual enterprises in dynamic reconfiguration scenarios, are better able than the existing transactional architectures.

We propose a communicational architecture able to achieve an effective integration of information systems, as well as designing its logical and functional model. We also define the necessary semiotic framework in order to a communicational integration architecture could be efficient and effective.

We implemented two prototypes to demonstrate the applicability of the proposed architecture. The demonstration of the research hypothesis was demonstrated with the realization of two experimentations where the ontologies have been unable to resolve disagreements or absences of opinion inherent in people who collaborated. This was overcome with the implementation of mechanisms that allow the co-creation between members of the group that participated in the trial.

This page was intentionally left blank.

Resumo

O objectivo desta tese é explorar Arquitecturas de Integração de Sistemas de Informação em condições de Reconfiguração Dinâmica de Empresas Virtuais. O principal desafio que identificamos e que serviu de base da pesquisa é que as tecnologias de informação por si só não conseguem suportar de forma eficiente e efectiva o conhecimento humano e a sua forma natural de interagir.

Já Sausurre (1916) defendia que parte do conhecimento residirá sempre na pessoa, e a tentativa de o tentar modelar é suficiente para que seja deturpado. E esse é o mote de todo este trabalho. Enaltecer as capacidades das tecnologias emergentes mas no sentido de elas permitirem a interacção homem-to-homem, sendo o sistema de informação meramente um meio para que tal seja possível.

Argumentamos por isso que uma arquitectura comunicacional de integração de sistemas de informação, onde Pragmatics mechanisms are enabled, em empresas virtuais em cenários de reconfiguração dinâmica, são mais capazes que as actuais arquitecturas transacionais.

Propomos para isso uma arquitectura comunicacional capaz de conseguir uma integração efectiva de sistemas de informação, assim como desenhámos o seu modelo lógico e funcional. Definimos ainda o quadro semiótico necessário para que uma arquitectura comunicacional de integração seja eficiente e efectiva.

Implementámos dois protótipos capazes de demonstrar a aplicabilidade da arquitectura proposta. A demonstração da hipótese de pesquisa ficou demonstrada com a realização de uma experimentação onde as ontologias se mostraram incapazes de resolver discordâncias ou ausências de opinião inerentes às pessoas que colaboram. Tal foi superado com a aplicação de mecanismos que permitiram a co-criação entre os membros do grupo que realizou a experimentação.

This page was intentionally left blank.

TABLE OF CONTENTS

ACKNOWLEDGMENTS.....	V
AGRADECIMENTOS.....	VII
ABSTRACT	IX
RESUMO.....	XI
TABLE OF CONTENTS.....	XIII
INDEX OF FIGURES.....	XIX
INDEX OF TABLES	XXVII
INDEX OF CHARTS	XXIX
LIST OF ABBREVIATIONS	XXXI
1 INTRODUCTION	1
1.1 Framework and Motivation.....	1
1.2 Objectives	4
1.3 Project Structure	5
1.4 Contributions.....	7
1.5 Thesis structure	9
1.6 Note to the Reader	10
2 INTEROPERABILITY IN CONTEXTS OF DYNAMIC RECONFIGURATION	11
3 STATE-OF-THE-ART OF INTEROPERABILITY	17
3.1 Concepts.....	17
3.2 Enterprise Interoperability.....	19
3.2.1 Interoperability.....	19
3.2.2 Fundamentals.....	21
3.3 Patterns and Architectures Models for Systems Integration.....	22
3.3.1 Integration Patterns	23
3.3.2 Architectures Models	26
3.3.2.1 Client-Server Model	27

3.3.2.2	Peer-to-Peer Model	28
3.3.2.3	SOA Model	29
3.3.2.4	Cloud-based Model	30
3.3.3	Artifacts for Modeling and Integrating	32
3.3.4	Selecting Architectures	34
3.3.5	Remarks	35
3.4	Frameworks, Architectures and Methodologies	36
3.5	From Data to Information: Retrieving, Searching and Selecting Strategies	38
3.6	Ontologies Interoperability	39
3.7	Interoperability-Ready Architectures	43
3.8	Interoperability Support	46
3.8.1	Web Services	46
3.8.2	Semantic Web and Web Services	50
3.8.3	WCF	57
3.8.4	Interoperability on Cloud	60
3.8.4.1	Cloud Impact	60
3.8.4.2	Real-Time Supporting Technologies	62
3.9	Interoperability on Virtual Enterprises and Manufacturing	65
3.9.1	Virtualization in systems integration	66
3.9.2	Dynamic Reconfiguration	68
3.9.3	Ubiquitous Manufacturing	72
3.10	Interoperability in Tourism Business	72
3.10.1	The Open Tourism Consortium	73
3.10.2	Dynamic Tourist Packages: some contributions	73
3.10.3	Web “Tourism” Services	74
3.10.4	Open Tourism Initiative	75
3.11	Reflection and Conclusions	76
4	SEMIOTIC INTEROPERABILITY FRAMEWORK	81
4.1	Pragmatics	81
4.1.1	Semiotic Integration: much more than ICT	82
4.1.2	Collaborative behaviours and supporting technologies	84

4.1.3	Towards Human/User Interface alignment.....	85
4.1.4	From User Interfaces to User Experience.....	87
4.2	Pragmatics vs Pragmatic Web.....	88
4.3	Semiotics in Virtual Enterprises and Manufacturing	89
4.4	Semiotics in Tourism Business	91
4.5	Reflection	92
5	COMMUNICATIONAL ARCHITECTURE MODEL.....	93
5.1	Transactional vs. Communicational.....	93
5.2	From Transactional to Communicational architecture	97
5.3	Semiotic based Architecture	99
5.3.1	Pragmatic “renderer” component	102
5.3.2	Cloud Architecture towards Ubiquity Manufacturing	103
5.3.2.1	More than an innovative business model	104
5.3.2.2	ICT, Dashboards and Cloud Manufacturing.....	105
5.4	Communicational Architecture	106
5.4.1	Conceptual Model.....	107
5.4.2	Logical and Functional Model.....	110
5.4.3	Technological Model	113
5.4.3.1	Hybrid Cloud based.....	113
5.4.3.2	Structure.....	114
5.4.3.3	Pragmatics.....	119
5.4.3.4	Summary.....	120
5.5	Cloudlet Architecture Dashboard.....	121
6	VALIDATION OF THE COMMUNICATIONAL DIMENSION OF THE PROPOSED ARCHITECTURE.....	127
6.1	The proposed architecture validation framework.....	127
6.2	Synopsis	128
6.3	Research Methodology	130
6.4	Experimentation description.....	132
6.4.1	Part A – User Interface Description	133

6.4.2	Part B: User Experience	133
6.4.3	Part C: User Pragmatics.....	134
6.5	Collaboration 1-to-1	134
6.5.1	Support	134
6.5.2	Findings	140
6.6	Collaboration N-to-N	141
6.6.1	Dynamics	141
6.6.2	Monitorization.....	142
6.6.3	Development	143
6.6.4	Step 1 – Taxonomies	144
6.6.5	Step 2 – Ontologies Interpretation.....	148
6.6.6	Step 3 – Revision of the interpretation.....	150
6.6.7	Step 2 and Step 3 – Mapping of Concepts	151
6.6.8	Step 4 – Joint analysis of ontologies.....	160
6.6.9	Step 5 – Co-Creation of the new Ontology	161
6.6.10	Findings	169
6.7	Relevance to the Communicational Architecture.....	179
6.7.1	In Transactional architectures	179
6.7.2	For Communicational architectures.....	180
6.8	Conclusions	184
7	IMPLEMENTATION OF PROTOTYPES FOR THE PROPOSED ARCHITECTURE	185
7.1	Cloud Ubiquitous Manufacturing.....	185
7.1.1	Entities	186
7.1.2	Relational Model and Class Diagram	187
7.1.3	Used Patterns and Anti-Patterns	190
7.1.4	Components.....	195
7.1.4.1	A – Market of Resources Engine	196
7.1.4.2	B – Brokering.....	200
7.1.4.3	C – Pragmatics Engine.....	201
7.1.5	Web Portal.....	202
7.1.5.1	Resource	203

7.1.5.2	Broker	204
7.1.5.3	Dynamic Reconfiguration.....	207
7.1.6	Mobile Application	210
7.1.7	Data Integration using <i>Flat File</i>	211
7.1.8	Repositories.....	212
7.1.9	Data Serialization.....	214
7.2	Cirrus - Dynamic Tourism Service	215
7.3	Summary.....	221
8	CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH.....	223
8.1	Conclusions from the research	223
8.2	Main Contributions	225
8.3	Limitations and suggestions for further research	226
9	BIBLIOGRAPHY	231
9.1	AUTHOR'S BIBLIOGRAPHY.....	231
9.2	REFERENCES.....	232
10	APPENDIX A – PROTOTYPE DESCRIPTION	I
11	APPENDIX B – MATERIAL FOR THE EXPERIMENTATION	XIX

This page was intentionally left blank.

INDEX OF FIGURES

Figure 2.1 - Integration at all levels of the enterprises	12
Figure 2.2 - Traditional pipelined architecture for the analysis of terms. Results of each processing stage are used as input to the next processing stage in the order of application.....	13
Figure 3.1 - Enterprise Integration Patterns	18
Figure 3.2 - Communicational Patterns	18
Figure 3.3 - Client-Server Models.....	28
Figure 3.4 - Cloud Systems more than Cloud Computing.....	30
Figure 3.5 - Gartner's point of view for cloud trends.....	31
Figure 3.6 - Button "taxonomy" discrepancies.....	33
Figure 3.7 - Architectures Models.....	34
Figure 3.8 - <i>Podio</i> App Market.....	42
Figure 3.9 - Mozilla Plugins	42
Figure 3.10 - Second Life MarketPlace	42
Figure 3.11 - Services composition.....	49
Figure 3.12 - WebServices Model and Architecture	50
Figure 3.13 - Semantic Web on Web Services.....	52
Figure 3.14 - WebServices Coordination	53
Figure 3.15 - OWL-S sub-ontologies.....	54
Figure 3.16 - Ontology: ServiceDescription	55
Figure 3.17 - Ontology: ServiceProfile and author's Vcard	55
Figure 3.18 - Ontology: <i>HoteisLivres</i> ServiceModel.....	56
Figure 3.19 - Ontology: <i>ServiceGrounding</i>	56
Figure 3.20 - WebRTC Architecture	63
Figure 3.21 - Long Pooling model.....	65
Figure 3.22 - WebSockets model.....	65

Figure 3.23 - VE Topologies	70
Figure 3.24 - OTI Architecture	76
Figure 4.1 - Solutions archetypes are changing.....	87
Figure 4.2 - Web Conceptual Model.....	88
Figure 4.3 - Pragmatic Web as a negotiation tool	89
Figure 4.4 - Human-Machine-Human relation	91
Figure 4.5 - Technology Fidelity.....	92
Figure 5.1 - Modeling the transaction of states	94
Figure 5.2 - Effectiveness in communication mechanisms (media).....	95
Figure 5.3 - Manual schemas to get effectiveness in development process.....	96
Figure 5.4 - Traditional architecture – several applications need to be managed.....	96
Figure 5.5 - Communicational Architecture – Transparency and Ubiquity of services.....	97
Figure 5.6 - Transactional Multilayer Architecture	98
Figure 5.7 - Communicational Architecture with devices “abstracted” as semiotics instruments ..	100
Figure 5.8 - MVC and MVP/MVVM models	101
Figure 5.9 - Communicational Architecture where devices are Pragmatics Renderers.....	101
Figure 5.10 - Multimodal interfaces for multiple Client devices classes.....	102
Figure 5.11 - Effective Cloud based Semiotic Architecture.....	103
Figure 5.12 - Web 4.0 brings intelligence	106
Figure 5.13 - Context Diagram – Scope and Profiles.....	107
Figure 5.14 - Actors and Uses Cases.....	108
Figure 5.15 - Conceptual Domain Model	109
Figure 5.16 - Pragmatic Channels	110
Figure 5.17 - Components and Entities Logical Model.....	112
Figure 5.18 - External Systems Integration	115
Figure 5.19 - RIA Pattern	115

Figure 5.20 - WCF Cloud Services	116
Figure 5.21 - Technological Architecture - MVC/RIA Pattern.....	117
Figure 5.22 - Technological support	118
Figure 5.23 – Integrated Communications Channels	120
Figure 5.24 - Components Dependencies Diagram	121
Figure 5.25 - UMS Supporting Architecture.....	122
Figure 5.26 - Cloud-based broker: (a) Process Plan (b) Stereotype (c) Candidate resources (d) Spatial Data in cloud.....	123
Figure 5.27 - Cloudlet Architecture (a) Dashboards (b) Cloudlet (service) (c) Enhanced cloudlet (d) Cloudlet with pragmatics instruments.....	124
Figure 5.28 - Cloud and Communicational architecture enhance sustainable interoperability	126
Figure 6.1 - Adapted User Experience Elements	129
Figure 6.2 - Experimentation stages	131
Figure 6.3 - UMS UI used in the Experimentation	133
Figure 6.4 – O_A	135
Figure 6.5 – O_B	135
Figure 6.6 - User B's interpretation of O_A	136
Figure 6.7- User B's interpretation of O_A – formal graphical.....	137
Figure 6.8 - User B's interpretation of O_A – formal textual	137
Figure 6.9 - U_A 's validation of interpretation ontology by U_B	138
Figure 6.10 - Excerpt of co-created correct interpretation ontology – formal textual representation	139
Figure 6.11 - Co-created correct interpretation ontology– formal graphical representation	139
Figure 6.12 - (a) Two Information Fields or ontologies with a common part, and (b) Unified Information Field or ontology after pragmatics, co-creation, process.....	140
Figure 6.13 - A1 Ontology	145
Figure 6.14 - A2 Ontology	145
Figure 6.15 - A3 Ontology	146
Figure 6.16 - B1 Ontology.....	146

Figure 6.17 - B2 Ontology	147
Figure 6.18 - C1 Ontology	147
Figure 6.19 - C2 Ontology	148
Figure 6.20 - D1 Ontology	148
Figure 6.21 - Agreement Scale	149
Figure 6.22 - Co-created Ontology between $A1 \Leftrightarrow A2 \Leftrightarrow A3$	162
Figure 6.23 - Co-created Ontology between $B1 \Leftrightarrow B2$	163
Figure 6.24 - Co-created Ontology between $C1 \Leftrightarrow C2$	164
Figure 6.25 - Co-created Ontology between $D1 \Leftrightarrow D2$	165
Figure 6.26 - Co-created Ontology between $B2 \Leftrightarrow C1$	166
Figure 6.27 - Co-created Ontology between $B1 \Leftrightarrow C2$	167
Figure 6.28 - Co-created Ontology between $A1 \Leftrightarrow A2$	168
Figure 6.29 - Search of terms: a) Vocabulary Broker; c) Agents in Semantic Web Services.....	181
Figure 6.30 - Pragmatics and Ontology Brokering Service	182
Figure 6.31 - Applied Communicational Architecture	183
Figure 7.1 - Relational Model for Resource, Task and Communicational Channel	187
Figure 7.2 - Relational Model for Production Activity.....	188
Figure 7.3 - Class Diagram: Resources, Tasks and Channels	189
Figure 7.4 - Service Contract.....	190
Figure 7.5 - Public Resource Information.....	195
Figure 7.6 - Manufacturing Market of Resources.....	196
Figure 7.7 - MMR WCF Services API.....	199
Figure 7.8 - Communicational Channels.....	201
Figure 7.9 - Resources	203
Figure 7.10 - Resource Communication Channels	203
Figure 7.11 - Resources Tasks	204

Figure 7.12 - Filtering Resource	204
Figure 7.13 - Resource Filters: (a) General; (b) Sectors; (c) Details.....	205
Figure 7.14 - Distance and Others selection criteria.....	205
Figure 7.15 - Applied Distance Filter.....	206
Figure 7.16 - Detailed Resource information.....	206
Figure 7.17 - Web Chat with the owner of the resource.....	207
Figure 7.18 - Sequence of Production Activities	207
Figure 7.19 - Geographical representation of involved resources	208
Figure 7.20 - Any resource has their own communication channels	208
Figure 7.21 - Alternative resources.....	209
Figure 7.22 - Communication Channels for alternative resource	209
Figure 7.23 - InfoWindow for Resource: (a) General; (b) Occupation and (c) Communicational Channels	209
Figure 7.24 - Resources registration on the mobile application.....	210
Figure 7.25 - Repository Pattern.....	212
Figure 7.26 - Resources Repository Classes	213
Figure 7.27 - ChannelRepository Classes	213
Figure 7.28 - TasksResourcesRepository Classes.....	214
Figure 7.29 - Tourist Resource Information: Details; Classification and Communication Channels	217
Figure 7.30 - Prototype main Fron-End.....	217
Figure 7.31 - Tourist Resource Operations.....	218
Figure 7.32 - Tourism Brokering services (I)	218
Figure 7.33 - Tourism Brokering services (II)	219
Figure 7.34 - Tourist Resource Public Information	219
Figure 7.35 - Tabular Tourism Activity	220
Figure 7.36 - Georeferenced Tourism Activity	220
Figure 7.37 - Tourism Activity Reconfiguration	220

Figure 7.38 - Channel Resources in Tourist Resource Reconfiguration	221
Figure 7.39 - Dynamically alternative Tourist Resources Selection.....	221
Figure 7.40 - Model for Applications that adopt Communicational Architecture for.....	222
Figure 8.1 - Services Integration.....	229
Figure 8.2 - Services integrations and Infra-structures sharing.....	229
Figure 10.1 - Supporting Platform	I
Figure 10.2 - PhoneClient Resource Registration	III
Figure 10.3 - PhoneClient Resource Channels Registration	IV
Figure 10.4 - PhoneClient Keyboard Emulator	V
Figure 10.5 - PhoneClient Market of Resources Management	V
Figure 10.6 - XML Schema for XML validation – Graphical representation	VIII
Figure 10.7 - Resources List	VIII
Figure 10.8 - Resources Channels.....	IX
Figure 10.9 - Resource Tasks.....	IX
Figure 10.10 - Broker Filters	X
Figure 10.11 - Resources Selection (I)	X
Figure 10.12 - Resources Selection (II).....	XI
Figure 10.13 - Map representation of filtered resources	XI
Figure 10.14 - Map InfoWindow with resource data	XII
Figure 10.15 - Resource Communication Channels	XII
Figure 10.16 - Web Chat Channel	XIII
Figure 10.17 - Reconfiguration Manager.....	XIII
Figure 10.18 - Reconfiguration InfoWindows events	XIV
Figure 10.19 - Reconfiguration Alternative Resources Management	XIV
Figure 10.20 - Communication with Alternative Resource	XV
Figure 10.21 - Reconfiguration using the Map	XV

Figure 10.22 - Reconfiguration Map Resource Event.....XVI

Figure 10.23 - Reconfiguration with Map Resource Details.....XVI

Figure 10.24 - Reconfiguration Alternative Resource ChannelXVII

This page was intentionally left blank.

INDEX OF TABLES

Table 2.1 - Comparison of “traditional” enterprises and Virutal Enterprises in terms of Integration .	12
Table 3.1 - Interoperability Architectures and Models.....	45
Table 3.2 - Semantic Web Services Technologies.....	57
Table 4.1 - Semiotic Ladder	90
Table 6.1 - Some mappings and ambiguities	141
Table 6.2 - Step 1 resultant Taxonomies	144
Table 6.3 - Step 2 Agreements summary	150
Table 6.4 - Step 3 summary.....	151
Table 6.5 - A1 ⇔ A2 Mapping.....	152
Table 6.6 - A2 ⇔ A3 Mapping.....	153
Table 6.7 - A1 ⇔ A3 Mapping.....	154
Table 6.8 - B1 ⇔ B2 Mapping.....	155
Table 6.9 - C1 ⇔ C2 Mapping	156
Table 6.10 - D1 ⇔ D2 Mapping	157
Table 6.11 - B1 ⇔ C2 Mapping	158
Table 6.12 - B2 ⇔ C1 Mapping.....	159
Table 6.13 - Mapping Percentages Summary	160
Table 6.14 - Step 4 Summary	161
Table 6.15 - A1 ⇔ A2 ⇔ A3 Co-Creation.....	162
Table 6.16 - B1 ⇔ B2 Co-Creation.....	163
Table 6.17 - C1 ⇔ C2 Co-Creation.....	164
Table 6.18 - D1 ⇔ D2 Co-Creation.....	165
Table 6.19 - B2 ⇔ C1 Co-Creation.....	166
Table 6.20 - B1 ⇔ C2 Co-Creation.....	167
Table 6.21 - A1 ⇔ A2 Co-Creation	168

Table 6.22 - A1 ⇔ A2 Co-Creation Analysis	170
Table 6.23 - B1 ⇔ B2 Co-Creation Analysis.....	171
Table 6.24 - C1 ⇔ C2 Co-Creation Analysis.....	172
Table 6.25 - D1 ⇔ D2 Co-Creation Analysis	173
Table 6.26 - B1 ⇔ C2 Co-Creation Analysis.....	174
Table 6.27 - B2 ⇔ C1 Co-Creation Analysis.....	175
Table 6.28 - Mapping between A1 ⇔ A2 ⇔ A3.....	176
Table 6.29 - A1 ⇔ A2 ⇔ A3 Co-Creation Analysis.....	177
Table 6.30 - Experiment Results Summary	178
Table 7.1 - Service Controller/Message API Patterns implementation.....	191
Table 7.2 - Resource DataContract.....	191
Table 7.3 - Asynchronous Response Handler Pattern.....	192
Table 7.4 - Resource API Pattern: DataSet serialization to JSON	193
Table 7.5 - C# JSON Serialization.....	193
Table 7.6 - JQuery JSON asynchronous utilization	194
Table 7.7 - Real-Time Pattern Service	194
Table 7.8 - A Stored Procedure	198
Table 7.9 - <i>NewResource</i> method of the API of MMR.....	199
Table 7.10 - openTok Communication Models.....	202
Table 7.11 - Resource XML File.....	211
Table 7.12 - LINQ2SQL over XML data.....	214
Table 10.1 – XML Schema for XML validation	VII

INDEX OF CHARTS

Chart 6.1 - A1 ⇔ A2 Mapping results 152

Chart 6.2 - A2 ⇔ A3 Mapping Results 153

Chart 6.3 - A1 ⇔ A3 Mapping Results 154

Chart 6.4 - B1 ⇔ B2 Mapping 155

Chart 6.5 - C1 ⇔ C2 Mapping 156

Chart 6.6 - D1 ⇔ D2 Mapping 157

Chart 6.7 - B1 ⇔ C2 Mapping 158

Chart 6.8 - B2 ⇔ C1 Mapping 159

This page was intentionally left blank.

LIST OF ABBREVIATIONS

VE	– Virtual enterprise
VEI	– Virtual Enterprises Integration
EAI	– Enterprise Application Integration
UMS	– Ubiquitous Manufacturing System
XML	– Extensible Markup Language
SOA	– Services Oriented Architecture
OTI	– Open Tourism Initiative
OTC	– Open Tourism Consortium
ICT	– Information and Communication Technology
WCF	– Windows Communication Foundation
REST	– Representational State Transfer
EU	– Europe Union
CSP	– Cloud Service Provider
IAI	– Inter-Enterprise Application Integration
API	– Application programming interface
SIP	– Session Initiation Protocol
XMPP	– Extensible Messaging and Presence Protocol
BOSH	– Bidirectional-streams Over Synchronous HTTP
RTC	– Real Time Communication
HTTP	– Hypertext Transfer Protocol
TPL	– Task Parallel Library
AJAX	– Asynchronous JavaScript and XML
FTP	– File Transfer Protocol
BBS	– Bulletin Board System

IRC	– internet Relay Chat
ETTL	– Extract Translation/Transport and Load
EIP	– Enterprise Integration Patterns
P2P	– Peer-to-Peer
VoIP	– Video over Internet Protocol
UDDI	– Universal Description Discovery and Integration
WSDL	– Web Service Description Language
PaaS	– Platform as a Service
IaaS	– Infrastructure as a Service
SaaS	– Solution as a Service
CPU	– Central Processing Unit
QoS	– Quality of Services
IDL	– Interface Description Language
RIA	– Rich Internet Applications
UIDL	– User Interface Description Languages
VDM	– Vienna Development Method
VDM-SL	– VDM Specification Language
XAML	– Extensible Application Markup Language
MDE	– Model Driven Engineering
EM	– Enterprise Modeling
ERP	– Enterprise resource planning
CRM	– Customer Relationship Management
TM	– Transactional Middleware
MOM	– Message-oriented Middleware
PM	– Procedure Middleware

LIST OF ABBREVIATIONS

OOM	– Object-oriented Midlleware
SOM	– Service Oriented Middleware
ORB	– Object Request Broker
ESB	– Enterprise Service Bus
CORBA	– Common Object Request Broker Architecture
DCOM	– Distributed Component Object Model
CBA	– Component-Oriented Architectures
ACID	– Atomicity, Consistency, Isolation, Durability
BPEL	– Business Process Execution Language
EDA	– Event Driven Architecture
B2B	– Business-to-Business
C2B	– Consumer-To-Business
BsC	– Business-to-Consumer
C2C	– Consumer-to-Consumer
BPEL4WS	– Business Process Execution Language for Web Services
OWL-S	– Ontology Web Language for Services
URI	– Uniform Resource Identifier
SOAP	– Simple Object Access Protocol
WSMO	– Web Service Modeling Ontology
JMS	– Java Message Service
MSMQ	– Microsoft Message Queuing
XSD	– XML Schema Definition
SSE	– Server-Sent Events
GAO	– United States Government Accountability Office
DTP	– Dynamic Packet Transport

CTAS	– Collaborative Travel Agent System
MAIS	– Multi-Agent Information System
SP	– Stored Procedures
CLR	– Common Language Runtime
UDF	– User Defined Functions
MVC	– Model View Controller
MMR	– Manufacturing Market of Resources
TMR	– Tourism Market of Resources

1 INTRODUCTION

This chapter presents the global considerations about this dissertation. It is structured in six sections: the first, which frames and describes the motivation that sustains this research; the second, which presents the main objectives intended to be achieved; the third, which describes the project structure; the fourth which presents a synopsis of the main contributions; the fifth with the structure and organization of this document and finally, the sixth with a note to the reader of this document.

1.1 Framework and Motivation

“In FP7 we built the means to understand, in FP8 we need to build the means to change”

Europe Commission

One of the trends or consequences resulting from the globalization of markets and business processes will be the increasing cooperation or collaboration between companies in the entire lifecycle of a product. The companies lose the traditional loyalty of customers and suppliers and must react quickly to continuous changes in the market. This requirement for flexibility will impact at all company technological (new applications, updates, etc.) or organizational levels (reorganizations, mergers, etc.).

All these changes bring new challenges and problems, putting the decision makers face to new strategies and decisions. With the evolving technologies, new possibilities emerge and the solutions, to continue to be useful, have to adhere to these new opportunities. It is necessary to capitalize prior investments (now seen as legacy solutions) without losing the rhythm imposed by the market. Interoperability is therefore a huge problem nowadays, since it is essential to keep the pace with high external rhythm but with minimum impact (cost, motivation, ability, etc.) on the internal one.

The socio-political context has established and will continue to be determinant on, after been promoted at the beginning of the decade with the *Treaty of Lisbon* (European_Commission,

2002), among others, the growth and sustainability of the economy based on science, knowledge and innovation; reinforced with the initiative *i2010 Strategic Framework* (European_Commission, 2005) that, when recognizing the importance of Interoperability for enterprises, promoted the alignment of the *Media* to support of an information society, establishing the global broadband (broadband) as the basic infrastructure for a modern economy and a society of real information, with the digital operationalization of all public services and "*along with the need to promote new, rich online content, increase interoperability between platforms and devices and raise trust amongst investors and consumers through enhanced security*" (European_Commission, 2009).

More recently the *JTI – Joint Technology Initiatives*¹, framed in *Building of Europe of Knowledge* of FP7² (FINES, 2009) program and now reinforced and sustained with the FP8³ (CORDIS, 2011), became the decisive step towards the promotion of global integration, when promotes the global alignment of R&D in areas considered critical to EU, through partnerships between public-private entities, involving companies, the scientific community and public authorities. One of these priorities is to combat the digital divide that separates, among others, business, companies and people. The current *Digital Agenda for Europe* (2010-2020, successor of i2010)⁴ clearly demonstrates the commitment of the EU, among others, in a convergent digital society, with mobility of resources and services, promoting the Internet as an engine of interoperability and standards application.

Thus, in this context of globalization, the business process (production, management, marketing, etc.) had to be reviewed and adapted. Being the competition now much more severe, global and critical, companies needed to focus on their *core business*, partnering with others, experts in the fields which does not dominate, but needs (Prahalad & Hamel, 1990)! This result in professional relationships with entities that do not know each other at all, unless the quality of the services it provides, in resorts of outsourcing services, subcontracting, etc. The actual image of the company is increasingly virtual, necessarily.

Following this behavioral and paradigm shift in perspective of the development of solutions, although the development teams (or production) will be composed of elements both internal and

¹ http://cordis.europa.eu/fp7/jtis/home_en.html

² 7th Research Framework Programme - http://cordis.europa.eu/fp7/home_en.html

³ 8th Framework Programme for Research and Technological Development in Europe

⁴ http://ec.europa.eu/information_society/digital-agenda/index_en.htm

external to the company, monitoring and coordination the work team must remain rigorous and made as a whole, and the management of information now coming from various sources and formats (external systems) becomes a more delicate task.

The company reorganizes into a collaborative way extended to partners, in an *Extended Enterprise* model. But to do so required advance study and stipulate rules and standards for data sharing, caring for issues such as security, privacy, etc.; for the integration of computer systems or even sharing of computer resources (emerging with virtualization and clouding) such as, network services, storage, applications, *SaaS (Software as a Service)* "applications", etc.; towards the standardization or definition of rules for process synchronization. That is, a set of assumptions and definitions able to make these synergies as transparent and efficient as possible, or, in another perspective, a set of meta information or *meta-computing data* (Schultheiss, Rijn, & Kamphuis, 2002), able to anticipating further difficulties in the operation of the entire integration process.

As the capacity is now distributed and almost ubiquitous, the need is to create the best networks of companies. This new virtual company is then a set of new participants that are able to participate and contribute to the same goal!

Identify and choose the best on a timeframe that does not compromise the use of the opportunity, demand these new organizations to reconfigure quickly and easily assimilate new requirements.

It is in this context of dynamism and multiplicity of entities (technologies, solutions, participants, objectives) that we are going to explore an architecture able to respond to the reconfigurability of virtual enterprises, that is required to be fast and as dynamic as possible.

One of the proposals is to apply the semiotic framework for information systems. This frame considers the information systems in six levels: 1 - Physical, 2 - Empirical, 3 - Syntactic, 4 - Semantic - 4, 5 - Pragmatic, 6 - Social. Given this, the "traditional" approaches include levels 1-4. Obviously, the pragmatic and social aspects are not treatable by mechanisms belonging to levels 5-6. The characterization of processes (software) at levels 5-6 is essentially communicational. The role of these processes is, in fact, the creation of new semantics as the dynamics of the needs, inherent to the processes of dynamic reconfiguration of virtual companies, i.e., inherent to the dynamic integration processes of heterogeneous information systems. This approach has led to the definition of the concept of *Generative Integration* (G.D.

Putnik, Cruz-Cunha, Sousa, & Ávila, 2005b), or "co-creation" (collaborative creation) and similar. For these reasons, we say that the integration approach that considers the aspects/levels 5-6, i.e. the pragmatic and social levels of the semiotic framework of information systems, is characterized by communicational processes.

In this way, the scientific and engineering issue is:

To develop the mechanisms/architectures for information systems integration of virtual enterprises that allow effectiveness and efficiency on that integration, in conditions of their dynamic reconfiguration.

1.2 Objectives

The integration of heterogeneous information systems belonging to the independent enterprises (which would integrate a virtual enterprise), is one of the problems, or critical success factors in the implementation of new organizational models, as Virtual Enterprises.

Thus, the central objectives of this doctoral project consist of:

- Specify models of integration architectures of information systems of Virtual Enterprises in terms of its dynamic reconfiguration, i.e. models of architecture based on transactional information systems, and a model of architecture based on information communication systems, such as a competing architecture;
- Develop software applications to demonstrate the effectiveness and efficiency of competing architectures in conditions of dynamic reconfiguration of Virtual Enterprises;
- Evaluate the effect of the architectures developed in the integration of information systems to networks of enterprises in sectors such as the Manufacturing and the Tourism ones.

These objectives are to characterize and understand the potential of integration architectures in situations of dynamic reconfiguration of inter-enterprise business networks (Virtual Enterprises) and develop software applications prototypes for demonstration of the features of the proposed architectures, in order to ensure the sustainability of this business model, with scientific support, implementation and validation of results.

The knowledge of the parameters of the potential of the proposed architectures and their technological capabilities for Virtual Enterprises sustainability is of utmost importance for the identification of the strategies to follow in the area of supporting technologies.

So, it should be demonstrated the following hypothesis (whose demonstration represents the scientific part intrinsic of this doctoral project):

The information systems integration architecture most appropriate in conditions of dynamic reconfiguration of virtual enterprises is the architecture based on communicational systems, contrasting with the 'traditional' architectures, based on transactional information systems.

The methodology for demonstration and validation of the hypotheses is based on laboratory experimentations and software applications developed for the purpose.

1.3 Project Structure

The project structure and work plan follow the five-stage project development model. The five-stages of this model are:

1. State-of-the-art analysis and definition of detailed project objectives;
2. Specification of an architecture model based on transactional information systems and an architecture model, based on information communicational systems, as competing architecture for integration of information systems of Virtual Enterprises;
3. Construction/development of a demonstrator for the architectures proposals;
4. Validation of the demonstrator for proposed architectures models;
5. Exploitation plan for the proposed architectures to support dynamic reconfiguration of Virtual Enterprises.

1. State-of-the-art analysis and definition of project detailed objectives

- Analysis of organizational characteristics and requirements inherent to Virtual Enterprises, especially in the point of view of their dynamic reconfiguration;
- Analysis of the features and functionalities of meta-institutions models as Virtual Enterprises integrators environment;

- Analysis of models, techniques and information systems integration architectures of intra and inter-enterprise based on transactional information systems;
- Analysis of models, techniques and information systems integration architectures of intra and inter-enterprise based on communicational information systems;
- Detailed definition of the project objectives.

2. *Specification of an architecture model based on transactional information systems and an architecture model based on information communicational systems, as competing architecture for integration of information systems of Virtual Enterprises;*

- Specification of an architecture of information systems integration based on transactional information systems;
- Specification of an architecture of information systems integration based on communicational information systems;
- Specification of a software application based on the architecture of information systems integration in transactional information systems;
- Specification of a software application based on the architecture of information systems integration in communicational information systems;
- Defining the properties of the proposed architectures;
- Identification of the conditions for application of the models and developed software;
- Development of an evaluation model and inherent application methodology;

3. *Construction/development of a demonstrator for the proposed architectures*

- Specification of the environment for implementation of the proposed model for the Manufacturing sector;
- Specification of the environment for implementation of the proposed model for the Tourism sector;
- Development of a software application based on the integration architecture of information systems of transactional information systems for application in the field of Manufacturing;

- Development of a software application based on the integration architecture of information systems of transactional information systems for application in the field of Tourism;
- Development of a software application based on the integration architecture of information systems of communicational information systems for application in the field of Manufacturing;
- Development of a software application based on the integration architecture of information systems of communicational information systems for application in the field of Tourism;

4. *Validation of the demonstrator for the proposed architecture model*

- Definition of specific interaction requirements;
- Application of the demonstrator;
- Hypothesis testing and validation;
- Discussion about the capacity and conditions of implementation of the developed architectures for use in the Manufacturing and Tourism industrial sectors.

5. *Exploitation plan for the proposed architectures to support dynamic reconfiguration of Virtual Enterprises*

- Definition of the fields of application of the proposed architectures;
- Presentation of the proposed models propagation scenarios in Manufacturing, Tourism and other sectors;
- Critical success factors in the implementation of the proposed architectures;
- Identification of the problems to be solved in the next phase of research and development

1.4 Contributions

The business opportunities arise unpredictably and for very short periods of time. The traditional enterprise behavior is replaced by the constant adaptation to new demands and offers of the

involved resources (technologies, partners, etc.), to tighter and global concurrencies, sustaining the market integration.

In the context of Virtual Enterprises this reconfiguration capacity is raised to extreme, where enterprises must be agile in decision-making, have to learn continuously (*Learning Organizations*) and to adapt to assimilate new requirements in processes in which they intend to participate (*Generative Integration*). Thus, the integration architecture that we believe to be necessary, as well as the framework that will be modeled, should be guided either by functional requirements, such as:

- Data persistence
- Resources Management
- Authentication and authorization
- Monitoring and *Logging*
- Managing Workflows
- Quick and dynamic reconfiguration
- Easy adaptation
- Evolving (learning capacity)
- Synchronous and asynchronous interaction
- Cognitive capacities in decision making
- Collaborative capabilities and negotiation
- Performance management

or not functional, such as:

- Characteristics inherent to an interoperability dynamic platform (scalable, flexible, extensible, etc.)
- Opened: based on standards
- Distributed
- Robust

It will be an architecture able to adopt integration capacities inherent to several models of architectures: *Component-Based Architecture*, *Model Driven Architecture*, *Event Driven Architecture*, *Service Oriented Architecture*, *Multi-Tier Architecture*, and *Agent-Based*

Architecture, i.e., seek to handle components, templates, services and agents, structured in multiple functional layers.

At the University of Minho, several PhD projects in the area of Virtual Enterprises have been successfully conducted. However, no project has addressed the more detailed specification of the various models of integration architectures of information systems for the conditions of dynamic reconfiguration of Virtual Enterprises. In this sense, this doctoral project will contribute to complete the range of research carried out at the University of Minho in this area.

1.5 Thesis structure

The dissertation is composed of 8 chapters (including the present one), five of which correspond to the five phases of the Work programme, as follows:

Chapter 2 describes the context and presents the problems inherent to the interoperability of dynamic reconfiguration of virtual enterprises. Also describes and distinguishes transactional from communicational architectures.

Chapter 3, after describing the main concepts related with integration or interoperability, it performs a state-of-the-art analysis on models, patterns, technologies and integration architectures for information systems integration intra and inter-enterprises. Enterprises networks and interoperability requirements, cloud interoperability and ontologies interoperability are also detailed contents. The interoperability of Virtual Enterprises and Manufacturing in scenarios of dynamic reconfiguration is explored.

Chapter 4 defines the semiotic framework of interoperability, exploring semiotic integration and the importance of Pragmatics to achieve it; the relevance of *User Interface*, *User Experience*, and the importance of communicational channels on the Pragmatics support is also considered.

Chapter 5 presents a model for the proposed communicational architecture. Being a semiotic and cloud based architecture, its conceptual, logical, functional and technological supporting models are described in detail. The potential of interoperability of the architecture is also emphasized.

Chapter 6 details the realized experimentation to validate the communicational dimension of the proposed architecture. The research methodology and the findings and its relevance to the communicational architecture are well described.

Chapter 7 shows in detail the implementation of prototypes for the proposed architectures. The Market of Resources engine Application Program Interfaces (API), the Brokering mechanism and the Pragmatic engine were the developed components. The developed Web Portal and the Mobile Application were applications that demonstrate the proposed communicational architecture application in Cloud Ubiquitous Manufacturing and Tourism economic areas, in their reconfiguration scenarios.

Finally Chapter 8 concludes the thesis, by remarking the main contributions and limitations of this project and identifying further possible research directions.

1.6 Note to the Reader

The reader should bear in mind that this text approaches two influenced but completely disjoint areas: Technology and Philosophy. The philosophical part arrives because at the core of information systems technological support lies a fundamental unknown: we are unable to define or delimit human knowledge in a formal or rigorous manner which allows for its full representation in computers (Brewster, 2008).

The technological base graduation on Computer Science of the author allows exploring with independence the essential relation between human and technology. A kind of paradox, the technology is made by and to humans, but, indeed, humans need to transform them self to use it.

Thus, the essential goal of the work focused on the need to harmonize the gap between the application services and the people that, to better relate, need and use it.

2 INTEROPERABILITY IN CONTEXTS OF DYNAMIC RECONFIGURATION

One of the fundamental differences between "traditional" enterprises, "monolithic", and virtual enterprises, i.e. enterprise network type, is the approach to the problem of system's reconfiguration.

A "traditional" enterprise may be regarded as a (relatively) stable structure, which tends, on the one hand, to avoid their reconfiguration because of costs, and on the other, also the integration into networks with other companies to protect their knowledge about the organization (management and technology) as a competitive factor.

The cost of reconfiguration, in both cases, i.e., in the case of internal reconfiguration and in the case of inter-enterprise reconfiguration, is called "transaction cost" (Coase, 1937) (Williamson, 1979). By contrast, the transaction cost and enterprise knowledge protection are inhibitors of networks creation/integration and their dynamics reconfiguration (Cruz-Cunha & Putnik, 2006).

Virtual enterprises "see" the creation of networks and their dynamic reconfiguration as a chance to improve or at least maintain its performance. To manage these objectives, virtual enterprises must have the specific mechanisms to eliminate these negative factors for networks establishment and reconfiguration.

One of these mechanisms, or rather, one of the groups of these mechanisms are effective and efficient mechanisms of integration of information systems oriented to the facilitation of dynamic reconfiguration of networks of enterprises that have, in principle, heterogeneous information systems.

Unfortunately, the state-of-the-art of information systems integration in virtual enterprises, although they have been made important contributions by the scientific community, is still far from a satisfactory state, with regard to requirements for the highest dynamic reconfiguration (G. D. Putnik & Cruz-Cunha, 2005a) (G.D. Putnik et al., 2005b) (G.D. Putnik et al., 2005b).

To better understand the problem, Table 2.1 presents the differences and the characterization of the integration between a "traditional" and a virtual enterprise (as a higher capacity network of dynamic reconfiguration).

This list of properties stems from the type of structure that supports a traditional enterprise. The limitations on interoperability (or integration) do not result only from software integration problems or information technologies, usually inherent to “resistant” legacy applications, but also from the different contexts of behavior of each enterprise, namely environments and business processes, organizational rules, competencies and knowledge

		"Traditional" Enterprise	Virtual Enterprise
1	Rental of integration	Intra-enterprise	Inter-enterprise
2	Organizational structure (project) vs. integration (process)	Decoupled	Coupled
3	Structural complexity (of organization)	Low	High
4	Volume/Number of relations of integration	Low	High
5	Dynamics on establishment of relations of integration	Low	High
6	Dynamics of integration processes	Low	High
7	Dynamics of needs for new integration mechanisms	Low	High
8	Generative Integration ^(*)	No	Yes
9	Integration life cycle	No	Yes
10	Language complexity	Low	High
11	Integration base	Transaction	Communication
12	Needs for the multi-dimensional approach	Low	High

Table 2.1 - Comparison of “traditional” enterprises and Virtual Enterprises in terms of Integration

^(*) including “auto-integration” as a model

The integration must occur at all levels of the company structure (Figure 2.1).

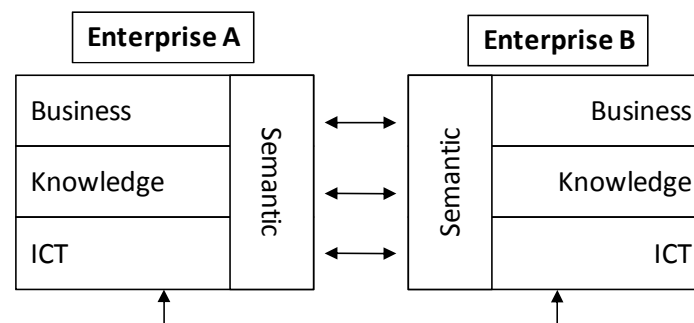


Figure 2.1 - Integration at all levels of the enterprises

One of the reasons for the non-existence of satisfactory solutions for integration of information systems of virtual enterprises is in the apparently not appropriateness of architectures/models/techniques of integration applied in "traditional" enterprises.

It can be said that the approach of the integration of "traditional" enterprises (in the scientific area of *EI-Enterprise Integration*), although following patterns of behavior already identified (centered on *EIP-Enterprise Integration Patterns*), is in most cases based on semantic tools, i.e. standards about data formats, ontologies, meta-data, and the like, which can be characterized as transactional information techniques. However, today we know that the information processing that covers the aspects of the languages up to the level of semantics (Figure 2.2), it is not at all sufficient. Thus, the solution must be found "elsewhere".

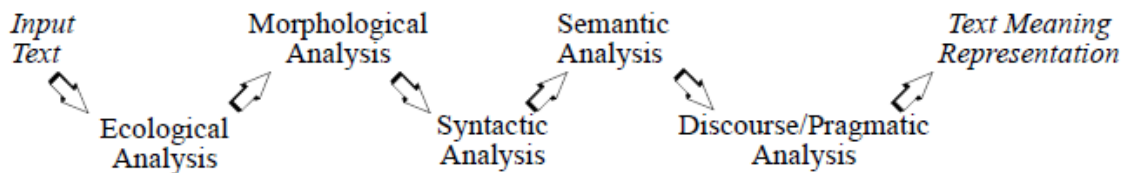


Figure 2.2 - Traditional pipelined architecture for the analysis of terms. Results of each processing stage are used as input to the next processing stage in the order of application.

(Bremer, 2008)

For instance, in the tentative to interpret some text which someone send to us, it is not difficult to constact that the interpreted message is not exactly the same of the initially intended sent message.

Agreeing with Bremer (2008), during the several steps of interpretation (Figure 2.2), many "personal" contributions exist when ambiguities or questions arise. Where syntactic (morphology) and semantic (meaning) questions exist, pragmatics (own interpretation and reasoning) solutions are used!

Paraphrasing Bremer (2008),

"The final result of the process of text understanding may include some information not overtly present in the source text. For instance, it may include results of reasoning by the consumer, aimed at filling in elements required in the representation but not directly obtainable from the source text. It may also involve reconstructing the agenda of rhetorical goals and plans of the producer active at the time of text production and connecting its elements to chunks of meaning representation."

Dynamic Reconfiguration

The Dynamic Reconfiguration or Reconfigurability Dynamics represent the more real of nowadays requirement for efficient and competitive enterprises. An enterprise needs to be reconfigured anytime that it is not able to follow customer's requirements or demands in a useful period of time. And reconfiguration means not only reorganizing spaces, methods or persons; it could means deeper transformations such us substitution of resources providers, generating new partners or cooperators, etc.

Citing Putnik (2005),

Face to the need of business alignment that the competition environment is demanding, enterprises are expected to present at least the following characteristics:

- *Fast reconfigurability or adaptability: the ability of fast change face to the unpredictable changes in the environment/market, implying the substitution of resources (i.e. the network structure can have as many instantiations as required either by product changes or as a requirement of quality and competitiveness improvement), and the*
- *Evolutionary capability: the ability to learn with history.*

These requirements implies the ability of (1) flexible and almost instantaneous access to the optimal resources to integrate in the enterprise; (2) design, negotiation, business management and manufacturing management functions independently from the physical barrier of space; and (3) minimisation of the reconfiguration or integration time.

Thus the capacity to ensure effective interoperability in several information systems of distinct enterprises cannot be possible only betting on technological parameters. Indeed, the technological framework is well sustained (a lot of standars, patterns, tools, etc.) and ensures efficient mechanisms to support the required integration, but only technological.

And in scenarios of dynamic reconfiguration, where quick decisions are essential, immediate "answers" are need and information systems could not support it at all. The experience of the people (information field) together with the context (time, economical, social and other) is as relevant as that automatic existing information.

If brokering mechanisms are essential to select the best alternative resources in reconfiguration scenarios, so the capacity to immediately "interact" with those resources is. This ability to "interact" represents their interoperability.

This interoperability frames technology based interaction (that covers syntactic and semantic integration) as well as pragmatics based interoperability, which could essentially mean human – to-human communication capacity. For instance, to have the possibility, if needed, to immediately communicate with the owners (person) of the resources by directly seeing and talking with him.

Resuming, the need of interoperability among all stakeholders (people, systems, resources, etc.) together with the pragmatics support, represent the base requirements to better support the dynamic reconfiguration of Virtual Enterprises, and thus the main goal to achieve with the communicational architecture proposed and modeled in this research.

3 STATE-OF-THE-ART OF INTEROPERABILITY

For the better perception of the characteristics and capabilities of an integration architecture to support dynamic reconfiguration of a virtual enterprise, it is essential to identify the requirements of the dynamism of these processes, identify and exploit the weaknesses of existing architectures based on transactional systems, as well as exploring the capabilities of the current and emerging information and communications technologies, and structural trends in behavioral changes on business processes and companies.

3.1 Concepts

In order to contribute to a proper taxonomy and to help clarifying the purpose of the research, it is essential to clarify the main concepts referred in the document, and common in the literature:

Integration: integrate effect or action; Merger of enterprises in different stages of the production process. Aggregation of parts to create and/or capitalize synergies among them.

Architecture: technological approach, scientific, cultural, etc. of construction or relationship of parts to an end.

Information system: Computing system able to operate a set of information.

Systems architect: Profile responsible for defining and coordinating the different parts of a system.

Integration architecture: architecture that supports an integration project.

Integration of information systems: In the computer world, integrate systems in some way (data, processes, applications, etc.) that supports the operationality, the tactics and the strategy of a joint system (according to EAI).

Enterprises Integration: or *Business integration:* communication, interaction and interoperability between two or more enterprises or companies.

Evolution of Enterprise Integration: in the beginning it integrates mainly data ... today integrates and coordinates business!

Virtual Enterprise (VE): 'enterprise' whose responsiveness is invariant to variations in the capacity of its support.

Virtual Enterprises Integration (VEI): aggregation of Virtual Enterprises to capitalize synergies among them.

Levels of Integration: structuring of the applicability of integration processes (according to EAI - Enterprise Application Integration).

Integration Patterns: Sustained procedural 'recipes'; guidelines applicable to integration processes.

Enterprise Integration Patterns: patterns for integration of enterprises (Figure 3.1)

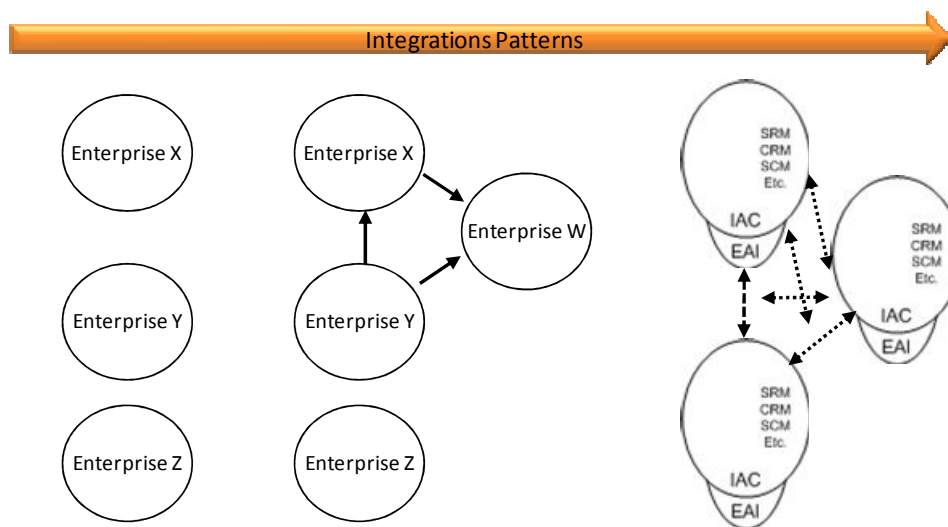


Figure 3.1 - Enterprise Integration Patterns

Communication Patterns: technological architectures that support communication processes (Figure 3.2).

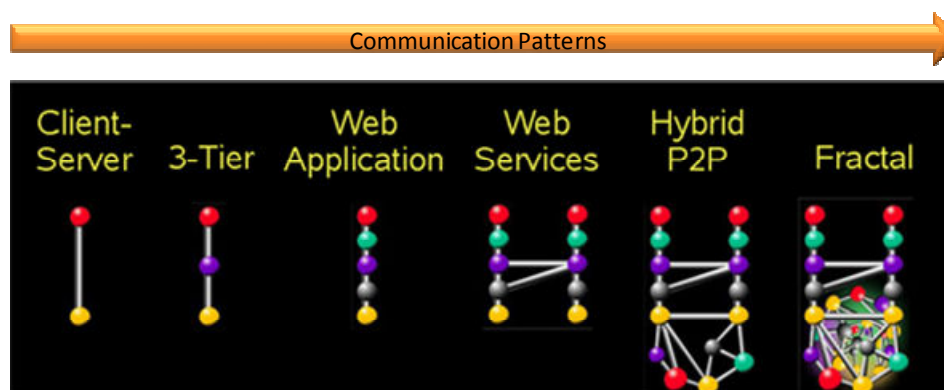


Figure 3.2 - Communicational Patterns

(Hansen, 2007)

Service: concept in the context of a company's business model that represents a business activity that can be performed.

3.2 Enterprise Interoperability

To well understand the real meaning of cooperation between enterprises, it is important to understand how can they be integrated and how the technology can facilitate that. In following topics we try to expose the basis of this process: the interoperability.

3.2.1 Interoperability

The ability of systems to provide and use each other's services effectively.

(Pokraev & Reichert, 2004)

The ability of two or more systems or components to exchange information and to use the information that has been exchanged.

(IEEE, 1990b)

"Ability to exchange functionality and interpretable data between to software entities"

(Mowbray & Zahavi, 1995)

Interoperability, integration, inter-cooperation, enterprises interaction, are common terms in the literature and, without risking to underestimate any of them and certainly others that have not been referred, in its generality, refer the capacity or ability of the enterprises to interact, based on technological platforms.

As Pokraev refered (2004), the IEEE definition of interoperability is in fact one of the most referred (IEEE, 1990a), but the framework presented refers to the interoperability of systems seen in the context of Information and Communication Technologies (ICT).

Since the interoperability between enterprises must cover the entire business context, including the support systems (information systems are just one example of several operating systems in the enterprise) and the context and processes of their businesses (M.-S. Li, Cabral, Doumeingts, & Popplewell, 2006), and for those definitions do not lack of objectivity, it is essential to understand those systems as a whole like a living system (biological), cooperative member (consumer, producer, etc.) on a "living ecosystem".

This divergence or lack of uniformity in the definitions is due to the continuous change of enterprises objectives, in turn inherent to the constant changes in demand, and typically

accomplice of emerging capabilities for communication of new technologies (Figay & Ghodous, 2009).

New capabilities and services promote new opportunities for interaction between systems, a fact evident with the integration and assimilation of the internet in the traditional business process, towards the *electronic business* (*eBusiness*, *eCommerce*, *eManagement*, etc.).

The integration of enterprises is not based only on technological issues. On the contrary, any level at which a company is structured will need a proper integration process with the same or different levels of the company with whom it intends to interact.

The vast majority of business applications, were not developed thinking in a future interoperability with other, even if semantically similar. Despite of this, much of the effort made to reach such capacity is based on models of one-to-one integration, i.e., each application is a particular case, using specific *Application Program Interface* (API) and follow integration patterns essentially at the level of the data. The integration will have to be made necessarily at multiple levels: at the level of application integration (physical, syntactic, semantic), which includes all communication requirements; at the level of business processes and even at inter-enterprise coordination level (Zwegers, 2005).

The attempt to use standards can safeguard interests of each participant but nonetheless are necessary tools (middleware) that map their own technologies in the chosen standards. The XML (*eXtended Markup Language*) is an example and represents almost a standard (by its widespread use) of mapping between data or data schemas.

It is noted that although there are developed many integration solutions within an enterprise or between enterprises, until the moment it has not been possible to develop a model seen as a standard or universally acknowledged.

Being still relatively short the history of the processes of information systems integration (hereafter abbreviated by ISI), is however already vast the range of explored technologies. In the next topic we try to make a historical review of technological considerations, its peculiarities and importance.

3.2.2 Fundamentals

The vast majority of enterprises developed their systems based on the best technology (and technicians) at the time, almost always specific to solve a particular problem. The concern to care for future cooperation and information sharing between processes would not be the key issue. It is a business model with multiple systems, functioning as islands with each other, supported by multiple technologies, many of which are proprietary and deficiently documented. We are faced with a heterogeneous system whose dependency between the parties is "manually" supported!

The need to "join" somehow all these islands or eliminate some of them, is the foundation of Enterprise Application Integration (EAI) that, according to Linthicum (Linthicum, 1999), is summed up essentially to a set of procedural and technological principles, which seek to coordinate this challenge, i.e., a way to integrate applications and data in enterprises, promoting the automation of processes. We refer, for example, the integration of (alredy and indeed) old systems (legacy systems) with new applications.

Different companies, from different areas, will have to deal with specific concepts of their business domains. Of course, these points will be transferred to the computer applications that are developing. Thus, the development of applications that allow integration between them forces the System Architect to discover ways of interoperability and to define layers that may be "common" to multiple domains (Microsoft, 2002), namely in the functional part of the systems. We are in area addressed by EAI, but at a larger scale, the *Inter-Enterprise Applications Integration - IAI*.

Nowadays the computer applications can no longer be isolated. Companies need to access and manipulate all the information easily, quickly and transparently. All applications (programs, data bases, etc.) are integrated into a global solution, essentially using a layer – middleware – that ensures interoperability between all parts.

So happen with the integration. Today the key order is web. Almost "everything" runs around the web and with everything that relates with it. We refer to development of IT solutions, the business processes, the government initiatives, research projects, spreading news and contents, etc. The letter "e" appears as a prefix to more and more areas: eGov, eHealth, eLearning, etc., which represents the adjacent web support.

But it is also remarkable the unstoppable progressive and "confusion" on the web. Using the web as a tool for work is delicate, complicated and even without the sure to be efficient. It proliferate multiple types of information, from multiple sources, but there is no guarantee of authenticity and quality of certification of the published content. And security issues are not all.

Eventually it lacks credibility and so generates confusion, as Brendan Eich said "Web working is a mess" (Brendan, 2008). Any web user, professional or merely curious, is confronted with a variety of applications, configurations, standards, rules, etc. that necessarily leads to confusion. A normal web search (via Google, Yahoo, etc.) can become a nightmare...

But it is clearly true that the line separating the useful from the useless, the secure from sensitive, is increasingly blurred and is referred to judiciously by most users!

If on the other hand we look at globalization of what is business process, in a Information Society ever more advanced to the Web, where economic transactions are made between suppliers and customers who no longer need (nor possibility) to know, where the loyalty between solutions and customers begin to disappear, where the "neighbour" of yesterday looks unknown but powerful developer today, where the important is the product or service, then the scenario is almost virtual and is well more dantesque (Ferreira & Putnik, 2008)! To add disorder and lack of credibility of information, with business processes and others that it depends on, it is a recipe tendentially explosive!

The social part inherent to people, admittedly complex, emerges as the latest variant to condition all this. We try to respond with a "new" web, the Social Web (Web 2.0, 3.0, etc.) in order to minimize its impact. The ubiquity of the computer is evident (with cloud). But increasingly tends to virtualize the human presence on both sides (Pettey, 2007).

3.3 Patterns and Architectures Models for Systems Integration

During next chapters will be explored the main Integration Patterns and Architectures for systems integration in the research related context.

3.3.1 Integration Patterns

As there is no single solution, over time will be experiencing practices and models, assessing and promoting its acceptance as a standard to follow. The *Enterprise Integration Patterns* (EIP)⁵ brings together a set of processes that represent the best practices to develop integration processes, to the point of creating a vocabulary and a graphical notation to represent all stakeholders and ways to interact in each of the types of integration.

In (Bates, Freeman, Freeman, & Sierra, 2004) is referred that the creation of a *pattern* represents a consensus and a proposal properly structured, documented and supported by successful practices, applied in the analysis processes and development of solutions. Thus they cannot result from a mere idea or invention but a sustained evidence of their applicability and constitute a "drawing" for a given problem, refer all specification details for that problem and propose a possible solution that would ensure some quality in the system to develop. Considering that applications are autonomous and heterogeneous, an integration process can be sustained by the following criteria (Microsoft, 2004):

Need: If an application does not need to interact with another, it can continue alone!

Grouping: it can be critical if the applications are grouped and too much dependent (*Tightly Coupled Applications*), either by hardware, by technology, by releases, by time, by people, etc., i.e., a small change in one of the applications involved may condition the applicability of all other.

Simplicity: it is very important that the integration process is simple, i.e., properly identified and with source code that avoid the need to change the applications involved. The acquisition of a new or modification of an existent application should be, on the one hand, possible and fast and, on the other, should generate a minimal impact on the whole system.

Technology: the integration of several different technologies will require different and specific interaction layers. Therefore require the specific know-how, appropriate training and eventually, in the worst-case scenario, the recruitment of persons/specialized services. The use of tools that support could mean the "be depend of" the company that developed them.

Data format: If the embedded applications require "exchange" data, it is necessary to develop processes type ETL/ETTL - *Extract Translation/Transport and Load*⁶, the ensure data

⁵ EIP – <http://www.eaipatterns.com/>

⁶ http://en.wikipedia.org/wiki/Extract,_transform,_load

conversion between the parties. It is essential to be aware of how data can evolve ...For instance the recent bet on XML is proof in the change of direction in this criterion.

Temporality: the sharing of data between applications (files, databases, etc.) should be limited to the minimum by the time factor, both in that "spent" in the process of "exchange" as the changes that the data might have. All matters relating to time (latency, timing, etc.) must be properly studied during the development process.

Synchronism: how a process depends on the other, i.e., should or not wait for the finish of precedes process? There are contexts in which such is not the case (*Asynchronous*), and others in which that is essential (*Synchronous*).

Features: applications may not interact only through the data. It may be important to integrate processes, i.e., an application needs to invoke a process of another, local or remote.

In practical and technological terms, the results of an integration process are reflected mainly on three components of applications: on Front-Ends, Data, Processes and Methods.

In (Hohpe & Woolf, 2003), it is possible to follow the entire process of identification and composition of different patterns of integration, in solving a real problem. It is also possible to verify that human involvement in these processes are referring only to the user profile that uses the system and adapting to it.

Depending on how the systems are supported and how they support the tasks that must perform, the patterns which govern them will also be different. If you are involved in a service architecture we can have *Services Architecture Patterns* and *SOA Patterns*; if we have an application that needs to support real-time operations, we will have *Real-Time Design Patterns*; if we have efficient web applications, we could have *Web Application Design Patterns* and whether the services are hosted in the cloud, *Cloud Patterns* should be taken into account.

The applicability of patterns is usually associated with a set of services or functionalities that is intended to implement. Here are some examples of services and related patterns that could be behind the communicational architecture modeled in this research:

Needs (Daigneau, 2012)	Pattern
How can clients execute remote procedures over HTTP?	<i>RPC API</i>
How can clients send commands, notifications, or other information to remote systems over HTTP while avoiding direct coupling to remote procedures?	<i>Message API</i>

How can a client manipulate data managed by a remote system, avoid direct coupling to remote procedures, and minimize the need for domain-specific APIs?	<i>Resource API</i>
How can a web service provide multiple representations of the same logical resource while minimizing the number of distinct URIs for that resource?	<i>Media Type Negotiation</i>
How can a web service provide access to internal resources like database tables, stored procedures, domain objects, or files with a minimum amount of custom code?	<i>Datasource Adapter</i>
How can web services with different APIs reuse common domain logic while enabling both synchronous and asynchronous request processing?	<i>Command Invoker</i>
How can development tools acquire the information necessary to use a web service, and how can the code for Service Connectors be generated?	<i>Service Descriptor</i>
How can a web service API reflect its clients' needs while enabling evolution and avoiding breaking clients?	<i>Consumer-Driven Contracts</i>
How can one simplify manipulation of request and response data, enable domain layer entities, requests, and responses to vary independently, and insulate services from wire-level message formats?	<i>Data Transfer Object</i>
How can a client avoid blocking when sending a request?	<i>Asynchronous Response Handler</i>
How can direct consumer-to-implementation coupling be avoided (Erl et al., 2008)?	<i>Contract Centralization</i>
You want to efficiently notify a set of clients that relevant data has changed (Douglass, 2002).	<i>Observer Pattern</i>
Efficient Maintainable Web App Development (Ver http://webdesignpatterns.org/category/categories/patterns).	<i>MVC - Model View Controller</i>

Cloud Integration Patterns

But this context of multiple services (applications, systems, etc.) required efficient interoperability to get it useful. But “*cloud applications and platforms are not very valuable unless they can reuse the critical corporate data that is typically locked away in various on-premise systems*” (Talend, 2011), and this clearly delays the cloud adoption. According to Forrester Research, “integration challenges with other applications” is cited as the second highest barrier to cloud adoption.

This hybrid and complex environments require stringent mechanisms of interoperability where on-premise and off-premise applications need to be efficiently integrated and coordinated.

“(...) as business solutions evolve to embrace cloud computing, it is only natural that integration solutions must change as well. The old approaches to integration based on centralized hubs no longer suffice. The elasticity, ubiquity and extensibility of the cloud demand a new class of integration solutions that must traverse corporate and geographic boundaries and must be able to adapt to changing business needs. Forward-thinking organizations choose integration solutions that can not only address the needs that cloud computing presents today but also adapt to future challenges (...)” (Talend, 2011).

From literature and pragmatically, the main characteristics of integration for cloud should be reduce to *Elasticity, Ubiquity, Extensibility, Security* and *Reliability*. Talent (2011) suggests that “Integration solutions should be open, modular and easy-to-use, to provide the greatest return on investment to the organization”. Furthermore we defend the need to complement this definition with the capacity to abstract technologically to ensure platform independence. Following the same reasoning, many interoperability questions are not only technological.

3.3.2 Architectures Models

Since the beginning, technical changes promoted new application models. From initial atomic instructions to actual services, physical technology (hardware) and functional technology (software) are always synchronized. From initial computer architecture (multiple sections: user, application, operating system and hardware) to actual Services Oriented Architecture (layered or multilevel) it is obvious the advantage in developing solutions organized in to small and simple parts, instead of complex and heavy modules. Initially there were multiple single applications and now they deal with multiple interoperable (and complex) services.

Faced with limitations, the critical parts are forced to overcome its limitations. Otherwise they are replaced by new standards, patterns or processes. From successive experience feedbacks, several approaches to structure these “parts” in multiple levels of abstraction – architectures - have emerged.

These patterns came from initial all-in-one desktop applications (1st generation), adapted then to support web applications (2nd generation), where each component can be developed by distinct entities and seamlessly integrated in one solution. Actually, it is experimented its cloudsacle (3rd

generation?) to support SaaS (Solution as a Service) applications, i.e., all parts are implemented as a working independently service and available for cloud API exploration (Microsoft, 2009). In summary, the actual architectures are based in a combination of several architecture models and not limited to a single one.

Since we intend to innovate existing architecture models to support semiotic-based models (G. D. Putnik & Putnik, 2010a) using pragmatics based collaboration tools, it is important to analyze how it could be done and how to distribute this requirement for existing layers. However, any architecture approach must grant its dynamic reconfiguration and inherent agility, scalability, performance and security.

Client-Server, Peer-to-Peer and Service Oriented Architecture are the most relevant multilevel architectures.

3.3.2.1 Client-Server Model

Client-Server architecture (Figure 3.3) appeared to improve flexibility, usability, scalability and interoperability since it relies on messaging services for communications between clients. Besides its great advantages (security, administration, etc.) the scalability and reliability is compromised by server capacity, indeed. Even it can support real communication between clients, it hardly supports (many) new agents or different clients (distinct platforms) (Erl, 2005). However, due to centralization of data (flat files, databases, etc.) some particular features can be well supported with client-server based solutions. Searching mechanisms is an example.

Although it seems old, the client-server architecture is still responsible for interoperability relations between server and clients applications (Raney, 2009). In current internet services, for instance, the FTP (File Transfer Protocol) sharing file “tool” is a client-server service. Processes that demand distributiveness, trust, synchronism and transaction support are here granted as happens with heterogeneous databases data migration, legacy systems interoperability or even heterogeneous clients interoperability (multiple devices, for instance) (Microsoft, 2009).

Considering interoperability (between clients), in the beginning it was support mainly by instant messaging communications based. After email service results and proof (email was born before internet, in 1965) solutions like IRC (internet Relay Chat) on eighties and BBS (Bulletin Board System) on begin of nineties, started giving their first steps in real time discussions, news sharing or exchanging messages and bulletins (Peter, 2004). Nowadays high supporting technology

allows these services to support also real-time video and others efficient communicational features.

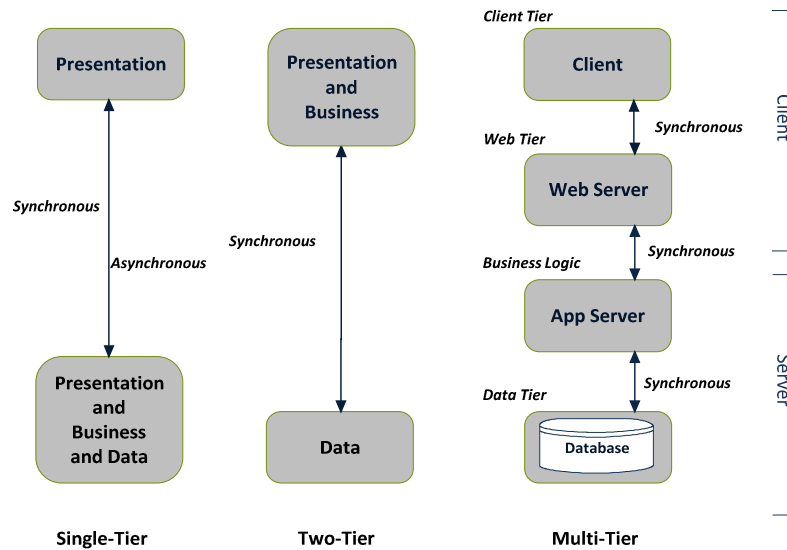


Figure 3.3 - Client-Server Models

3.3.2.2 Peer-to-Peer Model

On sequence of this architecture a new network (P2P - Peer-to-Peer) promoted direct relations between participants (web surfers), ignoring platforms, dismissing central coordination and easily sharing a lot of new resources. These participants (peers) need to provide also their resources (content, data storage, CPU, bandwidth, etc.) and not only demands (as happen with client/server). As new members join the network, the reliability, capacity and robustness increase, since there are more resources and lesser downtimes. Since any peer can send or receive contents they behave as supply (server) or consumer (client). In accordance to what Tim Berners-Lee (2008) said about web, P2P is an active “web” of links (peers).

The initial P2P specification focused the constitution of a high flexibility, dynamically, scalability, autonomy and high resilience network where each peer can join or leave easily and transparently be replaced by another. However the requirement to use proprietary tools to integrate P2P networks and heavy bandwidth usage, asymmetric bandwidth, dynamics of sharing (if peers want to act only as clients and never as servers), conflicts with ISP, etc., contributed to isolated the initiative (Olmedilla & er, 2005; Paul, Pan, & Jain, 2010).

The “overcome” of this handicap, allows this technology to be essential to support the needs of interoperability between systems and persons. It evolved from the initial file-sharing purpose, as happened with applications Napster (in 1990) and qBittorrent (in 2009) to important communication and interaction tools. Due to their communication capacities and VOIP (Video Over Internet Protocol) support features as⁷: Instant Messaging, File Transfer, Desktop Sharing, Voice and HD Video calling, Multi-party conference, Chat Rooms, Sessions Recording, Whiteboard, Extension Mobility, Call Transfer, etc., these kind of P2P applications become useful in multiple areas and for multiple purposes, as entertainment, politic, marketing, education, business, etc. One can refer Skype, Facebook, Google Talk, Windows Live, etc., as examples of these applications.

However with the emergence of social software (Figure 3.7), P2P technologies took new directions and actually peer-to-peer dynamic is in-vogue⁸, having computers and humans as peers. It involves peer production, governance and property, and gives emphasis to relational dynamic between participants. They behave equitably and with equivalent capability to participate and collaborate on a common good. The peers collaborate in the production and management and the results (property) are freely accessible.

3.3.2.3 SOA Model

Although the business opportunity is more frequent now, the reaction to it needs to be more effective. The developers need now to convert their applications to allow the interoperability between distinct potential (and technological) stakeholders. The (autonomous) services emerge as the solution to overcome the tightly technology dependence and their publication, discovery and composition are now the new challenges and tools to create new “applications”.

According to IBM's Phaedra Boinodiris (2010), SOA & Web 2.0 Product Marketing Manager, “Enterprise leaders and IT managers can exploit their enterprise's potential quickly with collaboration tools that overcome information and business process ‘silos.’ SOA revolutionizes not just the way applications work with each other but the way people interact with processes. Web 2.0 builds upon SOA's vision to support enterprises, foster Business/IT alignment and make companies more agile by offering a platform for services to be accessed, mashed & tailored” and

⁷ http://en.wikipedia.org/wiki/Comparison_of_VoIP_software

⁸ http://en.wikipedia.org/wiki/Peer-to-peer_%28meme%29#Infrastructure_for_social_P2P_processes

sustains also that “these Web 2.0-enabled experience networks empower people to collaborate, co-create, and experience personalized business processes as never before.”

With enhanced collaboration, using social networks and social computing, it is possible to create better conditions for “knowledge” or value creation (chat rooms, teamwork, etc.). Allowing the customers and suppliers working together on new technologies exploration and new solutions projections, the co-creation is completely enabled. And having the possibility to be involved of the product conception, their own preferences can be considered from the beginning.

SOA can grant services publication and interoperability, because it uses UDDI and WSDL standards. However cannot grant their availability or ubiquity (Coppinger, 2007), because that depends of the reliability, scalability and flexibility of its supporting infra-structure. Cloud services came to overcome this handicap since these characteristics are inherent to this new model of distributed architecture. Therefore manufacturing as a traditional dynamic business activity must be integrated on this “infrastructure” just to explore the advantage of its capability (G. D. Putnik *et al.*, 2011).

3.3.2.4 Cloud-based Model

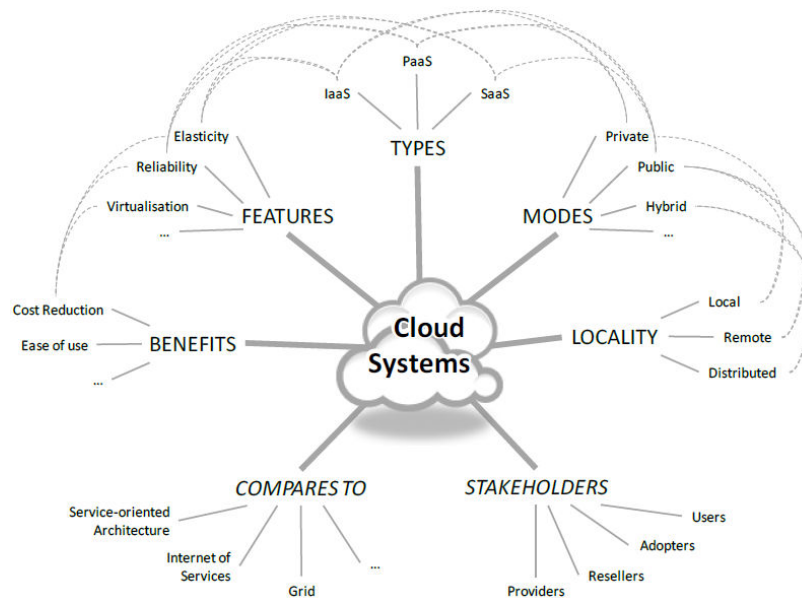


Figure 3.4 - Cloud Systems more than Cloud Computing

(Group, 2010b)

Effectively we defend that talking about cloud is not exactly the same as cloud computing. Cloud computing started from hardware questions (availability, scalability, security, etc,) questions, essentially. And in nowadays, cloud means much more than only infrastructures. Everything that

works on it, such as architectures, models, applications, process, systems, controllers and integrators must be considered too. The European Expert Group Report (Group, 2010b) stresses and forces the same point of view (Figure 3.4). Even Gartner has a definition for cloud computing only based in IT: *“a style of computing where scalable and elastic IT-related capabilities are provided as a service to customers using Internet technologies.”*

Its quick emergence is mainly due to SMBs (Small-to mid-sized businesses) since they got a great opportunity to get cloud-based applications and software-as-a-service (SaaS) (Figure 3.5), which are “easy to use, manage and provision, and they offer a “pay as you go” pricing model” (Talend, 2011), and thus a way to reduce costs and gain flexibility in their IT infrastructures.

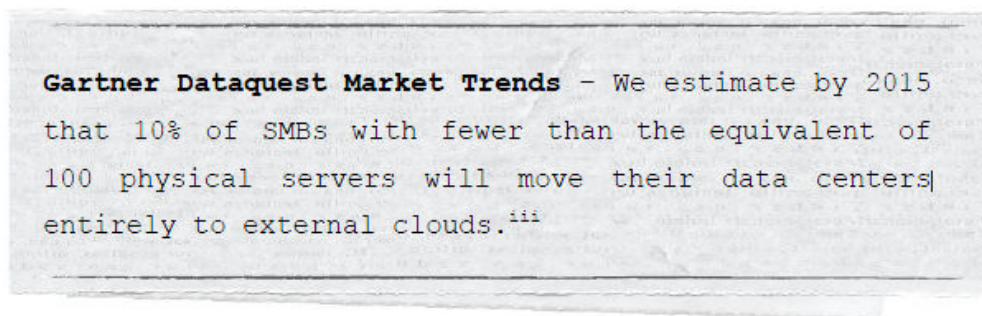


Figure 3.5 - Gartner's point of view for cloud trends

(Talend, 2011)

The companies can now be focused in their core business and “forget” questions related with infra-structures, for instance. That part is a service which is ensures for someone other company. Thus, companies believe that cloud today isn't about technology but services instead, i.e., *“...technology-enabled business models and business innovation...”* (Fingar, 2009). The companies need services (emailing, searching, shopping, collaborating, etc.) and not be cared with underlying technologies, security, availability and other “traditional” concerns.

There are essentially four types of cloud architectures: a) *consumer clouds*, where users are totally dependent of internet (cloud) services (provided by *CSP – Cloud Services Providers*), such as data sources, hardware, software, etc.; b) *public clouds*, where users participate as a consumer of a general public service (P2P applications, virtualization, SaaS, etc.), c) *private clouds*, essentially related with virtualization of data center resources. Public services are not allowed in the architecture is managed and is operated only by an organization, and d) *hybrids*, a union of more than one type of previous cloud architectures.

Also different models of cloud applications are described:

- a) *(Cloud) Platform as a Service (PaaS)*, which offers computational resources (as a framework) to allow the development or hosting of cloud based services. This development is based in the use of the specific APIs offered by those resources. Google App Engine, Microsoft Windows Azur are examples of it;
- b) *(Clouds) Software as a Service (SaaS)*, which represents already implemented services, business functions or processes, prepared to be used by (integrated on) other applications (or systems). Google Docs is an example of this;
- c) *(Cloud) Infrastructure as a Service (IaaS)*, which, indeed, represents resources or virtual “processing” capabilities, such as CPU processing, bandwidth, storage space, etc. *Amazon Simple Storage Service (Amazon S3)* is an example of this.

But the mixing of this different architectures and models represent the main common cloud based solution.

3.3.3 Artifacts for Modeling and Integrating

Decades of software development allowed Software Engineering to define (well) the necessary steps to efficiently implement a new software application. Being the study, analysis and specification (first steps) considered the most important (Sommerville, 2000) development phases, the user interfaces design still continue to be accepted as the easy way to show the alignment between functional system and client requirements. However, to take advantage of the emergent device’s interaction and collaboration capabilities, new applications must have multimodal user interfaces (Repenning & Sullivan, 2003) and implement Rich Internet Applications (RIA) features (Deitel & Deitel, 2008).

With XML and Services Oriented Architecture (SOA) the traditional heavy application were restructured in multiple heterogeneous (distinct source and technology) components (services) (Erl, 2009). Although legacy applications could be maintained, new business models demanded agile and loosely coupled applications that can respond quickly to continuous business requirements changes. Being this agility mainly supported by web services, the need to discover and integrate (or compose) them represented the new challenge and new IDL and *Universal Description Discovery and Integration (UDDI)* were developed for that purpose (Najdawi, 2009).

Despite of strengths of interoperability and reusability of SOA architecture, it has been penalized on multiple critical details, being the Quality of Services (QoS) and Security two of the most relevant (Hostetler, 2009). Multiple SOA implementations, large number of services domains and their not so easy interoperability implementation, were seriously compounded with cloud computing and their virtual infra-structures (Mulholland, Daniels, & Hall, 2008).

Due to domain complexity and heterogeneity, ontologies and taxonomies for services appear again as a possible solution to facilitate SOA adoption, improving the alignment between business (or domain) and information technologies. Having found in literature multiple scientific initiatives on SOA ontologies development (S. Cohen, 2007; Workgroup, 2010), and many projects focusing on their integration (Torniai et al., 2011), prove the complexity to get defined an unique ontology.

Considering UI, the scenario looks even more complex, since there is some “confusion” between *UIDL - User Interface Description Languages* and UI ontologies (García, Calleros, Vanderdonckt, & Arteaga, 2009; Paulheim & Probst, 2010).

Ferreira (2005) explored this complexity when applied formal methods (VDM-SL) to unify existent UIDL (UIML, XIML). Recent initiatives (XAML - Microsoft, Flex - Adobe, YUI - Yahoo, XUL - Mozilla, etc.) focus components description mainly, not concerned with their interoperability. In fact they are open markup languages that allow distinct description for the same component (Figure 3.6 shows examples of these discrepancies on *button* descriptions).


	
<code><Button Content="Click Me"/></code> XAML	<code><mx:Button id="button1" label="Click Me" /></code> Flex
<code><button type="button" value="Click Me"/></code> YUI	<code><button value="Click Me"/></code> XUL

Figure 3.6 - Button “taxonomy” discrepancies

So, the challenge to compose (reuse or integrate) UI components shall continue be technologically complex, obviously. Even “technically” different, looking carefully to Figure 3.6, one can clearly see that all describe the same!

3.3.4 Selecting Architectures

Moving from one initiative to another is basically motivated by some inefficiency of the first option. New potentialities and opportunities require quick adaptation and dynamic reconfiguration based on continuous upgrades or acquisitions.

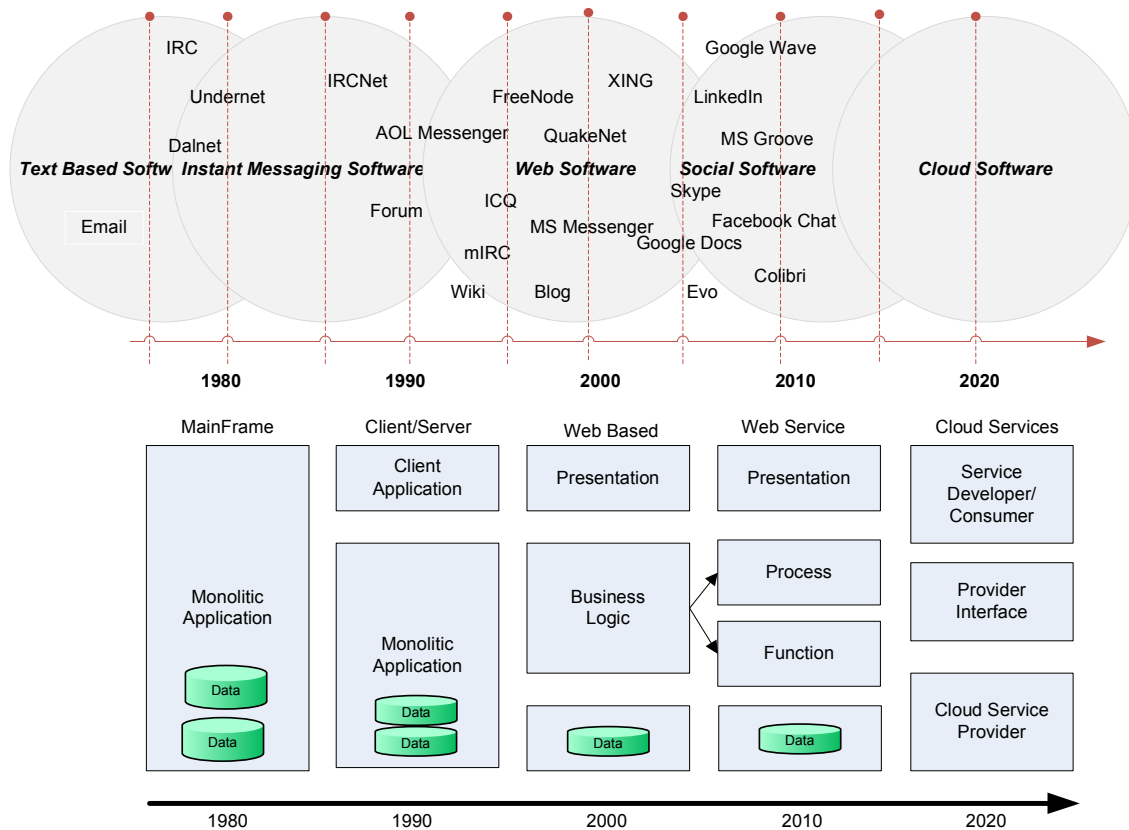


Figure 3.7 - Architectures Models

Several adaptations (Figure 3.7) considered technical details and integration or interoperability requirements. According to this “evolution”, the applications changed also to address new technology potentialities. Independently of that, applications behave mechanically: they process received data and output results (Wiehler, 2004)

Services were the main focus that reoriented last five year’s applications development (ITIL.org, 2011). Since services oriented architectures to support were designed and new business rules are now possible to be supported. Services can be published, discovered and composed. Applications result from scratch developments, components integration or from integration (composition) of existent services.

With cloud services modeling architecture the applications can easily behave as cloud service (SaaS) and supported in cloud infra-structures (IaaS). To implement them the traditional IDE were now substitute by platform as a service (PaaS) as is the case of Google App Engine.

With new architecture new usage patterns appear. Focusing on Manufacturing example, it can have a) a private part (Manufacturing System Service, for instance), b) a public part (Payment and Shipment services) and c) a community part (Manufacturing Inventory Service), towards a) a private cloud, b) a public cloud and c) a community or federated cloud.

Although it seems efficient, these models of algorithm based computing, created to improve efficiency and minimize the human dependency in the decision making, are not sufficiently functional to those who will use it. The user interpretation of the context is difficult to transmit to the system even he used more sophisticated devices.

3.3.5 Remarks

All architectures models and patterns are valid, exist and still continue evolving. At minimum to support and to capitalize legacy systems or processes. However, when comparing them, it is important to not compare only on the technological perspective.

Being supported by web services, SOA behaves as a strategy to quickly develop new business adapted solutions (Group, 2010a). However there are a lot of services that can be used, so, to discover “the best one” (best means quality? performance? whatelse?) that suits the purpose is not an easy task. To complement its discovery and “deal with” QoS (Quality of Service) difficulty, it is necessary an infra-structure which offers lower provisioning cost, better reliability, self sustainability and ubiquitous access. The cloud, behaving as a technology independent initiative, appeared exactly to support services execution with enhanced features. In a complementary perspective, cloud computing appear as a new development paradigm towards an efficient exploration of cloud services potential (Mulholland et al., 2008).

However there is unanimity that cloud (its computer power, platform and services) another multilayer architecture, must consider the combination of several others architectures and models where client/server and P2P technologies belongs too (Mulholland et al., 2008).

3.4 Frameworks, Architectures and Methodologies

Frameworks and Architectures are many times treated as the same concept, but in the majority of cases identified, a framework follows a particular architecture or model. Basically correspond to an implementation of a particular architecture, based on a set of several technologies or properly integrated and cooperating systems.

The scientific contribution in this area⁹ is vast and very difficult and complex to describe objectively since each Framework was designed and developed with specific purposes. The idea here, however, is try to identify frameworks and their methodologies adopted that applied in interoperability, and analyze the perspective to associate them to the purpose they may or not be a model to be applied in dynamic reconfiguration of virtual enterprises.

The difficulty in interoperability between enterprises (even more accentuated when virtual enterprises) is essentially conceptual, technological and organizational incompatibilities (Chen, Dassisti, & Tsalgatidou, 2005). According to (ATHENA, 2006), the European proposal of *Framework Programme FP6*, interoperability must occur at all levels of the company depends on behavior, i.e., data, services, and business processes. Clearly checks here not contemplation of semiotic and pragmatic issues.

From multiple frameworks identified in the literature review, we highlight some of those that we think they have some technological potential to apply to the agile behaviors of virtual organizations. They sustain on patterns and are supported mostly by technology of *agents* and *services*, or *Business Process Modeling* methodology, technologies (and methodologies) that we consider with potential to support the expected reconfigurability.

The *ExPlanTech*¹⁰, a framework based on open technology architecture, multi-agent, component-based oriented (ProPlanT), reconfigurable and flexible, supporting distributed applications and dynamic data management, enables the integration of agents and meta-agents that complying with the standard of interoperability FIPA¹¹ and promotes an ontology for semantic interoperability. Intended to be an alternative to support decision making in production processes, it can supports agents or components for real-time control, for instance.

⁹ <http://www.iso-architecture.org/ieee-1471/afs/frameworks-table.html>

¹⁰ <http://ubiquity.acm.org/article.cfm?id=763742>

¹¹ <http://www.fipa.org/specs/fipa00086/XC00086C.html>

Figay (2009) already proposes a platform for interoperability of applications in VE sustained on the use of eBusiness standards. It aims to achieve a "pragmatic interoperability" when seeking to integrate the various frameworks used by the different VE and who already work upon these standards, utilizing the gains of the models *MDE (Model Driven Engineering)*, *EM (Enterprise Modeling)*, SOA, and the web.

A recent and technologically robust architecture, structured from previous models of integration (LISI, IOM, LCIM, MITRE, SOSI, IMM), however, and in our point of view, supports only a programmatic interoperability (Meyers & Smith, 2007) and not pragmatic, since it focuses only on the ebusiness area, following a set of *best-practices* identified throughout the work. Despite its complexity it will be a reference to explore better in the work to develop.

(Shen, Hao, Wang, Li, & Ghenniwa, 2007) proposed a robust framework of service-oriented integration, based on agents technology, able to coordinate the processes of producing in a network of virtual enterprises and intending to respond to intelligent collaborative production processes. Here the services agents (web services) "negotiate" the scheduling of the processes of a given production order. The character preferably synchronous reveals the applicability of this proposal more to cooperative than collaborative processes, the latter on the basis of *Learning Organizations*, desired behavior model for companies to integrate in reconfigurable networks.

They may also be highlight several other frameworks associated with the different levels and characteristics of interoperability, but that do not fall or technological issues (e.g. Orbst proposal, from MITRE organization), or process (example of SOSI, from Morris), or quality (e.g. ATHENA).

Already in the technological context, the PLANETS (Botana et al., 2007) demonstrates the need and ability to manage multiple technologies and functional requirements but as Obrst (Obrst, 2004) argued, the essential condition for a semantic interoperability lies in the existence of low technological dependencies (*loosely coupling*), asynchronous communications and expressive semantic representation of what you want to integrate.

(Cordeiro & Filipe, 2003; G. D. Putnik & Putnik, 2010b; Yeh & Nason, 2004), proposed already interoperability frameworks to match the semiotic challenges, which analysis should be essential material for this work.

3.5 From Data to Information: Retrieving, Searching and Selecting Strategies

The common frameworks and technological support, focuses the knowledge discovery, its interpretation and preparation to offer the best answer to who needs it, as one of the main concerns.

Since the mobile commerce (m-commerce) is prevailed, economic activities (as Manufacturing, Tourism, etc.) must to adapt to this new channel of information, requiring new processes, marketing strategies (S. Liu, 2005). Considering the example of tourism activity and paraphrasing (Ferreira & Putnik, 2008), the “possibility to get useful information depends on the capacity to retrieve, search and interpret it. Considering this and accepting tourist information ubiquity, the actual mobile tourist profile looks real and mobile devices should be the key tool for information retrieval”.

In order to strengthen the relevance of the context, many others variants must be considered, namely, temporality, user preference, user experience, geographical information and pragmatics. Regarding tourism, the literature clearly evidences this:

- Kenteris *et al.* (2007) present personalized online tourism services; (Hill & Wesson, 2008) explores preference-based searching capacities to align searching results with tourist interests; and Lorenzi (2007) explores multiagent knowledge-based recommender system to deal better with disperse information and improve the consistency of recommended results; in (Barta, Feilmayr, & Grun, 2009), modularized ontologies show their capacities to model contextual information, towards semantic alignment between tourism service and user context and support better datamining.
- A service-oriented travel portal is being proposed to provide tourists with composite travel packages through dynamic composition among travel-related services from distributed providers and across business domains (Y. Li et al., 2011).
- The developments of Dinh & Thi (2010) are conducting to the development of a conceptual framework for service modeling in a network of service systems, based on network configuration and shared information.

- (Alptekin & Büyüközkan, 2011) are proposing a framework integrating case-based reasoning system with the Analytic Hierarchy Process multi criteria decision making technique to enhance the accuracy and speed in search and selection of suppliers in tourism destination planning.

Selecting and ranking several results using collaborative-filtering over previous similar experiences and making intuition on user's past behavior and user's stereotype similarities (Silvia & Amandi, 2009); doing knowledge-based inference on user needs and preferences (Middleton, Shabolt, & De Roure, 2004); applying case-based reasoning and multi-criteria decision making of (Alptekin & Büyüközkan, 2011); delivering relevant content to tourist under location-based systems (Schwinger, Grün, Pröll, Retschitzegger, & Werthner, 2006); data-mining over relational databases with online analysis processes (Chaudhuri & Dayal, 1997); integrating data using patterns and markup languages (Hohpe & Woolf, 2004); adapting context-based multimodal adaptive systems (Höpken, Scheuringer, Linke, & Fuchs, 2008); etc. are all well referred technical initiatives, essentially based on events and transactions and applied to concretes and objective scenarios.

Agents and web services (to enhance the discovery process), advanced matching algorithms (to enhance the accuracy and efficiency), case-based and context-related inference mechanisms define the generalized and relevant offer of the more recent scientific contributions. A hybrid combination of several of these technologies is the basis for most architectures and frameworks analyzed.

However, all these technical initiatives only can "infer" new information from existing and registered information or facts. The information which belongs to the user perspective is impossible to get before its manifestation (spoken, written, other) neither be effectively interpreted unless by another human. There still exist an important gap between the man and the machine communication.

3.6 Ontologies Interoperability

The attempt to provide interoperability suffers from problems similar to those associated with the communication amongst different information communities. The important difference is that the actors are not persons able to perform abstraction and common sense reasoning about the meaning of terms, but machines. In order to enable machines to understand each other we also

have to explicate the context of each system, but on a much higher level of formality in order to make it machine understandable (Brewster, 2008).

Tripathi et al. (2006) defend the use of ontologies as one of the dimensions of interoperability for an effective web portal, since “*Ontology is used to effectively combine data/information from multiple heterogeneous sources*”, and ontologies allow “*a shared and common understanding of some domain that can be communicated between people and application systems*” (Y. Ding, Fensel, Klein, & Omelayenko, 2002)

Although portability and interoperability between services had been promoted by SOA, the resistance to adhere to this paradigm shows that that was not so easy. Even known the domain complexity and heterogeneity, ontologies and taxonomies for services appear again as a possible solution to facilitate SOA adoption, improving the alignment between business (or domain) and information technologies.

Considering that literature presents multiple scientific initiatives on SOA ontologies development (S. Cohen, 2007; Workgroup, 2010), and many projects focusing on their integration (Torniai et al., 2011), prove the complexity to get defined an unique ontology; and considering that there are several research initiatives to map different ontologies using semantic relations, lexical taxonomies, text exploration algorithms, the results prove the complexity and limitations of these processes (Malucelli, 2006; Shahzad, 2011); after used ontologies to align business terminologies, the problem shifted now to ontologies integration, due to the quantity of domains and heterogeneity. In order to integrate several applications of distinct domains it is necessary to integrate their ontologies (Paulheim, 2011).

Ontologies interoperability (or integration) has been continuously discussed. Guarino (2004), saying that the “*lack of technologies and products to dynamically mediate discrepancies in business semantics will limit the adoption of advanced Web services for large public communities whose participants have disparate business processes*”, admitted ontologies integration discrepancies and proposed a solution: technologies.

Although the possibility to have collaborative ontology edition (ex. Protégé Editor¹², Ontolog¹³), and even known that it is a complex process (Tudorache, Noy, Falconer, & Musen, 2011), the main

¹²<http://protege.stanford.edu/>

¹³<http://ontolog.cim3.net/>

difficulty remains when ontologies need to be “compared” or, even more delicate, when someone wants to describe using its own terminology.

Social bookmarking inherent to nowadays social network applications, like *flickr*¹⁴ or *del.icio.us*¹⁵, represents a collaborative classification or tagging model, where anyone can classify its photos or other contents using any type of tag. However, this kind of free “tagged” ontology – named *folksonomy* by scientific community, even resultant from people experience, contributes for a global anarchy of taxonomies (Kim, Scerri, Breslin, Decker, & Kim, 2008). The need to develop an ontology to folksonomies is now evident (Knerr, 2006). Since the cycle of ontologies to “order” ontologies will never stop.

Although essential on semantic search mechanisms (Oliveira, 2008), the existence of large number of distinct ontologies demands analysis and designs patterns, and requires efficient mechanisms to retrieve ontologies from information (Hoekstra, 2009), to validate (Villela, 2004) and to use them on information sharing (Stuckenschmidt & Harmelen, 2005).

The scientific community is conscious about the scale of interoperability problems in consequence of emergent cloud development applications paradigm (Kuyoro, Ibikunle, & Awodele, 2011; Lawton, 2009).

Ontologies Interoperability essentially focuses their integration, understanding, composition or comparison. If ontologies are used to semantically “describe” cloud services, and if the Interface Description Languages (WSDL as example) are technical artifacts to describe and publish them, the services composition, condition *sine qua non for their interoperability, demands coherence between ontologies and IDLs (Paulheim, 2011). In practice, if an application (or component) is classified as a CRM service, we can integrate it in our ERP. Is it always possible?*

*Podio*¹⁶ application, for instance, appeared to support dynamic workflow workspaces, allowing the selection of several applications from an App Market (Figure 3.8). Even known that all applications can easily be integrated and used in Podio, and even well classified, their selection is not immediate, because there are many applications for the same purpose, and the interoperability between them does not exist.

¹⁴ <http://www.flickr.com/>

¹⁵ <http://delicious.com/>

¹⁶ <https://podio.com/>

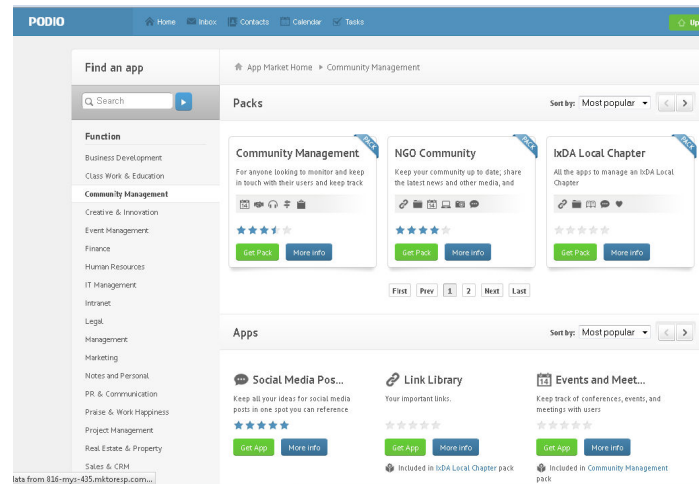


Figure 3.8 - Podio App Market

The same “problem” happens with many other applications or frameworks, being those add-ons, plug-ins (Figure 3.9), components, or others types, free or not (Figure 3.10).

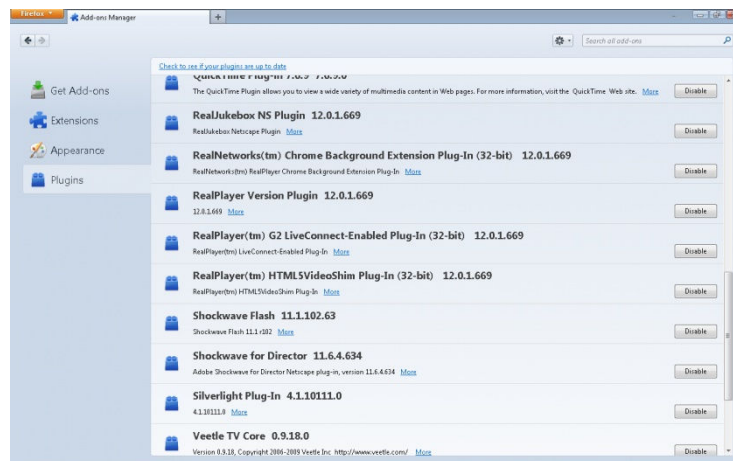


Figure 3.9 - Mozilla Plugins

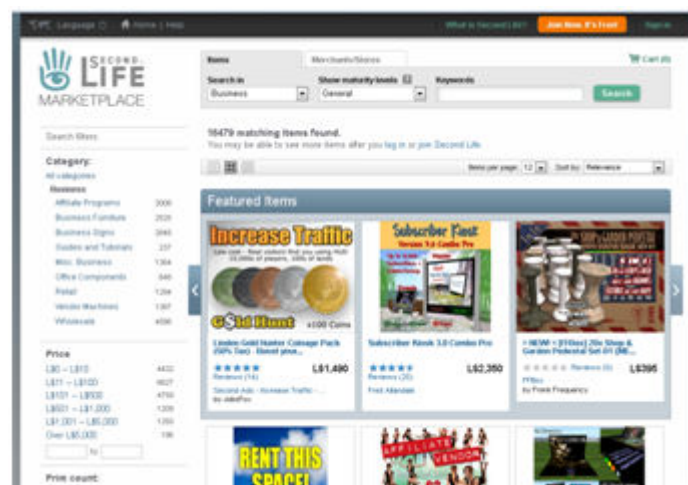


Figure 3.10 - Second Life Marketplace

In the particular case of Manufacturing that faces dynamic and global market rules, its Market of Resources (Cruz-Cunha & Putnik, 2006) with resources (services) prepared to work together (to be integrated) states nowadays ubiquity and agility requirements (G. D. Putnik, 2010; G. D. Putnik & Putnik, 2010b).

Effective interfaces between resources (services) states their interoperability, i.e., to compose an effective dashboard with distinct applications (services) to monitor a manufacturing process, the capability to make those components “talk together” is determinant.

In summary, independently of what layer one intends to integrate (front-end, business processes or rule, data), the capacity that each layer component has to interact with another one is essential. And ontologies are, in fact, a successful technological mechanisms to achieve that. But that is not sufficient.

3.7 Interoperability-Ready Architectures

Integration architectures, in the majority of cases and in functional terms, are focused on mapping (*middleware*) between existing systems and/or technologies.

There are multiple solutions exploring various middlewares type: *Transactional Middleware* (TM, TPM), *Message-oriented Middleware* (MOM), *Procedure Middleware* (PM), *Object-oriented Middleware* (OOM), and *Service Oriented Middleware* (SOM), i.e. solutions that rely primarily on transactions, messages, objects, agents and services.

The Web is now a mere support because next business will be on Cloud using services applications (Patrizio, 2010). In all of them it is necessary to know the specifications of the data (or schemas) source and destiny data for what is intended to transform.

In structural terms, were found by brokers supported architectures, some are based on messages (*Message Brokers*) (IBM, 2001), other on *Processes* (Business Brokers) (Johannesson & Perjons, 2005), where the broker ORB (*Object Request Broker*) promotes and ensures as mediator, the exchange of "content" between the objects; and other supported by Buses (bus), as is the case of ESB (*Enterprise Service Bus*) (Figay & Ghodous, 2009), which shows itself as one of the architectures of greater flexibility and robustness.

But the majority of these models require complex and little structured processes in modelling of the applications or processes to integrate, where the ability of abstraction is not always adequate

and where some proprietary technologies still insist on disrupting, compromising efficient flexibility.

The CORBA and DCOM are two examples of specifications of these architectures, which even accepted that are not proprietary specifications and although the promise of little (but complex) compatibility between them, both provide Interfaces and respective IDLs for remote access but, in fact, are technologically dependent (tightly coupled).

After a short passage through *Component-Oriented Architectures* (CBA) (such as the *ActiveX*), and the radical change of the business paradigm, where the web responds for most of the responsibility of the transactions, *Services Oriented Architecture* (SOA) promised to decisively contribute to the flexibility and scalability of applications.

Flexibility here refers to the ability to be able to use other capabilities without worrying too much about how this ability was implemented or supported. This ability translates into services (*Web Services*) implemented by others and which it is needed only discover and invoke. Shall then be possible to use these services regardless of platform, language, transport protocol and message format, ensuring *loosely-coupled* behaviors. Being an open specification and based on standards, almost any supplier can implements or supports it. This means that properties such as flexibility in the design of new services, reusing of existing services, interoperability and integration of existing and the easiness to to create new functional units composing other existing, are promoted eou guaranteed (Erl, 2007).

This possibility of composition and flexibility, along with technological independence are strong arguments to apply this architecture in environments of dynamic reconfiguration.

However SOA does not deal properly with dynamic services, with event-driven or critical transaction processes (ACID events - *Atomicity, Consistency, Isolation, Durability*)¹⁷, as it is more suited for processes request/response type (preferably synchronous) and requires great support for development due to its high degree of complexity. For example, some of the existing development platforms (*IBM WebSphere, BEA WebLogic, Microsoft .NET*) does not take care of interoperability between them.

¹⁷ <http://en.wikipedia.org/wiki/ACID>

Although there are SOA patterns via broker and *Web Services Orchestration* attempts (Mark Endrei et al., 2004), important issues are still unresolved, such as the semantics or even farther, pragmatics in Web Services (K. Liu, 2008).

In the process of enterprises integration, the questions will come to the point of being necessary to "integrate" different SOA implementations since it is supported by an enormous diversity of standards (*BPEL*, *ESB*, etc.) and too focused on processes of each business. SOA requires so something more and this means that enterprises have to look to what is happening "out there", reacting to the "events" generated by messages or stimuli that come from there.

In this kind of change of strategy, to become competent, flexible and agile, the development or business units of the company have to be insensitive to changes that are not coming from the "market".

Models		RPC	Client/Server	CORBA	DCOM	Tiers	REST	BPM	SOA	SaaS	ESB	MDA	EDA	WCF	P2P
Utilization	Open	●	●	●	○	●	●	●	●	●	●	●	●	○	●
	Proprietary	○	○	○	●	○	○	○	○	○	○	○	○	●	○
Complexity		□	□	⊕	⊕	⊕	□	⊕	□	⊕	⊕	⊕	⊕	⊕	⊕
Abstraction		□	□	⊕	⊕	□	⊕	⊕	⊕	□	⊕	⊕	⊕	□	□
Aplication	Services	○	○	○	○	○	●	○	●	●	○	○	○	●	○
	Objects	○	●	●	●	●	○	○	○	○	○	●	○	○	●
	Processes	●	○	○	○	○	○	●	○	○	●	●	●	○	○
	Models	○	○	○	○	○	○	○	○	○	○	●	○	○	○
Dependence	Loosely coupled	○	○	●	○	○	●	●	●	○	○	●	●	○	○
	Tightly coupled	●	●	○	●	●	○	○	○	●	●	○	○	●	●
Scalability		□	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Flexibility		□	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
Distributed		○	●	●	●	○	●	●	●	○	●	●	●	●	●
Real Time		○	○	○	○	○	○	○	○	○	○	○	●	○	○
		○ – No	● – Yes	⊕ – Very		□ – Little		⊕ – Significant							

Table 3.1 - Interoperability Architectures and Models

In this context it is argued that there should be an architecture that can protect the pace and direction of development, when changes occur in the company. This new architecture must then be based in the events and not directly on services. Obviously those services will be one of the generators of events. As business processes are standardized, their independence and operability will be ensured focusing only on demand. We are in the presence of an

architecture focused on events, *EDA-Event Driven Architecture*, by some called SOA 2.0 (Hoof, 2006).

So, if the context requires synchronous or transactional processes, the SOA model may be appropriate. In the case of processes with workflow defined, with asynchronous nature, as B2B processes perfectly dimensioned, etc., model EDA prevails.

We assume that this is another step in the shortest way to support the integration of VE in environments of dynamic reconfiguration.

Table 3.1 presents a significant set of the most common architecture models and their proprieties in software development able to be applied in the development of integration solutions. The distribution is based on the nature of the models, essentially.

3.8 Interoperability Support

This chapter details the most relevant technological approach on scenarios of dynamic reconfiguration of Virtual Enterprises. The key concepts are technology independence and infrastructures support for ubiquitous and reconfigurable systems. The web and the *cloud* are the kernel infrastructures.

3.8.1 Web Services

At a time when we question the initial objectives of web, the web services represent, in all its aspects, the latest point in the evolution of integration technologies.

Although we can understand that we are going back to the context of distributed computing, indeed we actually need to deal with something distributed. But in this time, beyond the processing capacity, emerge many other actors: there are processes, information, enterprises, persons, etc.

Assuming the ubiquitous internet in practically all forms of communication, also interesting is to use this infrastructure to support integration processes. A real example as an analogy, when making a call via cell phone to another destination, we are believers that, technologically, the call can be performed, regardless of the type of device that the target has. Similarly, a computer

should be able to communicate with another, regardless of their technical characteristics and processing capabilities

We must note that the web was initially designed for different purposes that perhaps justify some of the inefficiencies associated to it. From the beginning the idea was an infrastructure that would guarantee the fast and easy exchange of unstructured information. On the other hand, the first applications developed for the Web did not follow any standard or design pattern, initially as isolated and static applications (common *sites*), which could become outdated quickly.

This ad-hoc development promoted the emergence of services which interoperability could not be guaranteed. The applications were clearly designed to respond to human needs rather than to satisfy the "whims" of machines (Gokhale, Kumar, & Sahuguet, 2002).

Why so success?

Continuing in this analysis, the challenges of ICT press executives to try to reduce costs, innovate the infrastructure, improve service to customers, to be more competitive and to respond efficiently to the strategic priorities of the business (Wiehler, 2004).

Two main reasons can be deducted for this emerging reaction: the heterogeneity of systems and its applications and the rapid change of the market requests. Most companies dealing with multiple applications and multiple vendors, to meet multiple customers/requirements. To develop a unique and dedicated solution would be too expensive. But as we saw earlier, several different applications require increased efforts to interconnect them. On the other hand, the strongly competitive and dynamic commercial context forces companies to adapt quickly and thus adapt their ICT infrastructure. The pace of change clearly accelerates with globalization and becomes sometimes unpredictable in the direction that will be taken.

The global competition requires production cycles each time shorter, in order to get some advantage to other suppliers; the needs and demand of customers also change according to these short production cycles; the technology seeks to follow these new request "rhythms".

Considering this, it is easy to enumerate the main enterprises concerns:

- The inevitable integration of multiple technologies, with a strong risk of impact to the company and consequent high costs;
- Increased complexity of ICT infrastructure;

- Legacy applications to require attention in maintenance and customization;
- Dependence on suppliers of the acquired solutions;
- Process automation hampered either by the heterogeneity of the systems, whether by security implications, etc.;
- Difficulty in "open" collaboration to external *clusters* by inadequate infrastructure or even security issues.

As an example of all that, a new (electronic) business paradigm generalizes up and and installs itself clearly on the strategies of companies. Terms like *e-Commerce* (or eCommerce), *e-Business*, *Business-to-Business* (B2B) and *Consumer-To-Business* (C2B), *Business-to-Consumer* (BsC), *Consumer-to-Consumer* (C2C), among many others, preach that.

Essentially now is the quality of service that the supply can perform and not how he can support it.

The web services (or WebServices) are aimed at promoting a platform independence. An application should be "executed" regardless of operating system, hardware, etc., that the "target" possess and should be able to interact with others that are available via the Web (Mark Endrei et al., 2004).

*"A Web service is a software system identified by a URI whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols."*¹⁸

It is importante to understand a service as something different than a function, a method, a module, a program, etc. It should be understood as a modular and autonomous entity or even business logic unit, with its own domain, able to run something (like a PROM on a *chipset*) in reaction to the momentum of the data it receives.

¹⁸ Web Services Architecture Requirements, 2004, <http://www.w3.org/TR/wsa-reqs/>

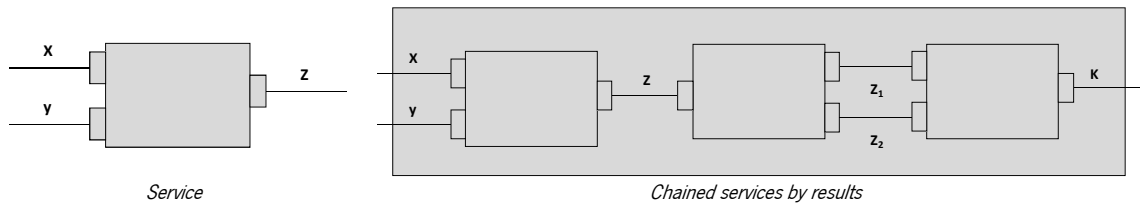


Figure 3.11 - Services composition

They work like pieces of a jigsaw puzzle. A final application may result from one or more chained and cooperative services (Figure 3.11). For example, you can "mount" a calculator by composition (chaining) of four other autonomous services: sum, subtraction, addition and division.

If in this combination of services, essentially remote, we use internet protocols (typically HTTP) as a way for invoking and "combine", and the XML to describe messages that are exchanged between them, i.e., able to be published, found and used in standard form and regardless of the platform, we will be in the presence of XML WebServices, commonly abbreviated only by WebServices.

Making an analogy with Objects or Components (in Computing Programming) we can say that:

- As objects and components, the services are actually chunks of code that allow you to implement certain functionality;
- As objects and components, the services combine information (e.g. status of classes) with behavior (e.g. properties of classes)
- Prevent the disclosure to the outside of the implementation details (e.g. encapsulation in object)
- Provide an interface to the outside (e.g. API)
- The objects use data abstraction. Services make it through the context or guidance aspect.
- Objects and components are structured into classes or hierarchies, services can also be used alone or combined with others.

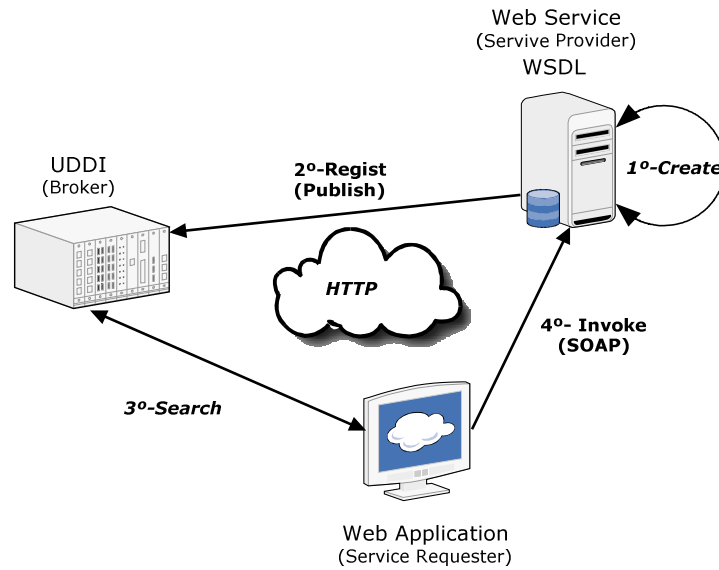


Figure 3.12 - WebServices Model and Architecture

So, the WebServices should be seen as a distributed middleware technology that leverages the simplicity and universality of XML as a way to allow the interaction of Web applications (not include here the common static websites). In essence, the applications that need services, do not need to know or worry about how they were made and in what language were implemented. All this complexity should be "hidden" by the infrastructure that supports these services. Just need to know if is there and where (Figure 3.12).

3.8.2 Semantic Web and Web Services

"The first step is putting data on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web — a web of data that can be processed directly or indirectly by machines."

Tim Berners-Lee, Weaving the Web, Harper San Francisco, 1999

We are now presenting a set of additional features that the services need to behave, in response to the emerging needs of "having to decide" and "have to assess". This is because the current context no longer cares about how they should or can develop Web services, but with the ability to select the best service in a so vast network of possible services.

The context

Since the Web was not designed to deal with rationalism, is today framed in an open space, filled with complex knowledge and organized in various formats (Yihong Ding & Xu, 2007). Thus it is necessary today additional tools to help the person to navigate in this space of knowledge, in order to obtain the best use of it.

By very intelligent or rational that we judge the systems are, it is still very difficult to remove the real meaning of the document (semantics), not the one who can interpret but the intended by the author who published it. The information that exists on the web have been placed to be read by people, not rationally and automatically analyzed by machines (Antonioni & Harmelen, 2008).

If we accept that the current Web search engines, such as Google, Yahoo, etc., with sophisticated search capabilities (such as Fuzzy, Euristics, etc.), are fitted with some intelligence capabilities, how to understand for example that, when searching for information about a particular term, arise a number of responses unaffordable?

Considering that the Web is used primarily for searching and integrating information and services, then the current Web, apart from the positive, ensures a set of problems. According to (Yu, 2007), this lack of ability to understand the real meaning of information leads to the occurrence of three major problems:

- The first is associated with the results of a Web search. A searching process results in a set of useless info or disassociated from the intended objective, making difficult to the user to figure out which is the more appropriate.
- The second is related to the implementation of web services. In the majority of cases they are processes with some complexity and require significant manual intervention.
- The third is related to the capacity and quality of doing *DataMining* on the info that is on the Web. This type of tools (usually very expensive) are programmed to "discover" a particular data type, and in contexts something accurate. It is difficult to manage to adapt to new contexts or other data types. Adaptability and flexibility are very limited. So the capacity is limited.

It is in this set of limitations of current Web that links the importance of complementing the information to be published and associated services, with some meta-information (document, content and relation) able to represent such a meaning that is needed. Understand with this the

need to insert any semantics on the Web – *Semantic Web* (T. Berners-Lee, 2001; L. Ding et al., 2005).

Semantics in WS

When we consider the Web services, preponderant vehicle in the current panorama of Web systems integration, either from the internet or an intranet, a description of the services, their location and their use will be enriched with a few more properties. The XML itself in which rests virtually all the technology that supports Web services, merely describes resources and ensure a syntactic interoperability, i.e., if the schemas between the parties are not known, it is difficult to "sense" each other.

On the other hand, as we develop services will happen what now happens with the information on the Web. There are too many services and with reduced aggregation and/or classification. Thus, the location of WebServices will become poor and surely we will return to a state that already is familiar.

Thus, while web services have already justified their credits to "migrate" the Web of a dispersed information base for a machine with large distributed computational power, there is however a perspective depletion of its current model and emerge the need of knowledge rules that still maintain it sustainable.

As the current applications begin to depend on the integration and cooperation of various services, to instill semantics in this process, we will rely on another kind of unusual integration: the semantics - *Semantic Interoperability* (Figure 3.12) (Sycara, Paolucci, Ankolekar, & Srinivasan, 2003).

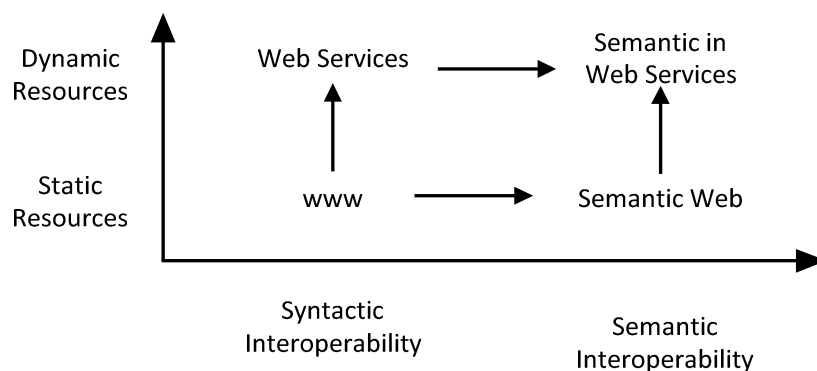


Figure 3.13 - Semantic Web on Web Services

When we insert semantics in web services, we frame some of the real current problems of this model, for example, the lack of a certificate of trust in the quality of the intended services. Very little is said about the quality of the services provided, unless the "register" of previous utilizations.

Imagine the following scenario:

"A tourist wants to spend their holidays in an instance of snow in the Swiss Alps. Need to book accommodation in a local Hotel, according to certain requirements, in particular, proximity, costs and housing conditions. After booking will be required that his GPS is "loaded" with the route from the airport to the Hotel, since the outward journey by plane was also reserved as a result of the process. As the journey to the hotel will be made by car, the Car-Rental company will be advised to put a car available to the client. All planned activities during the five days of holidays have already been fully prepared for him and rest of the family. Everything is planned and properly planned: travel (including return), accommodation, insurance, shipping, etc. It is possible to submit an expense report that will involve all the activity and the percentage of guarantee should be immediately charged."

The whole process was prepared remotely, via web, and with the minimum intervention of the person concerned, since the respective profile was made known (period, availability, preferences, number of people, etc.). Figure 3.14 seeks to represent the set of services involved in the preparation of this activity

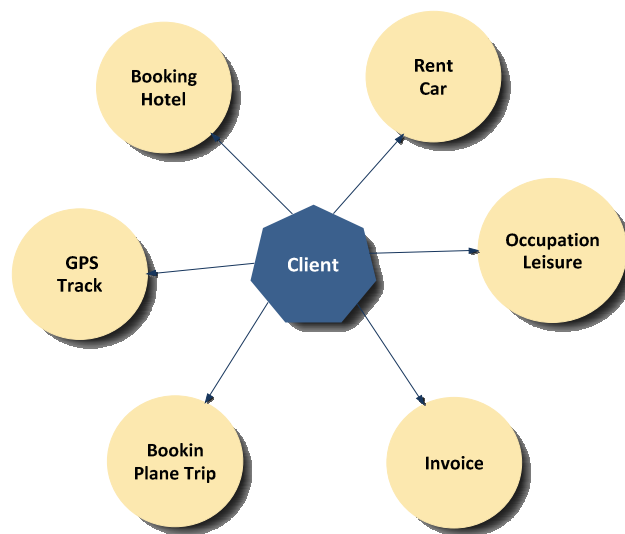


Figure 3.14 - WebServices Coordination

Easily one can check that are involved several services that must be coordinated and synchronized with the rules and requirements to meet. Assuming that these services are supported by web services, all of them should have, in addition to all the features involved, a set of meta-information to assist in decision making. Is clearly involved some semantics in decision-making for the proposal, this achieved through semantic Ontologies (set of concepts and rules to manage the relationships between them. In practice are models of how things should work) (Obrst, 2004).

This meta-information should "enriches" the service description (WSDL) and its location (UDDI). For that it is necessary to use other ways (as does the BPEL4WS¹⁹ in the area of *e-Business*) that permit, in particular the web ontology *OWL-S – Semantic Markup for Web Services* ou *Ontology Web Language for Services* (more expressive than *RDF*, *RDF-Schema*, *OWL* and *DAML*) (Martin et al., 2004).

The basic idea of this ontology is to describe for each service: *what does the service do?* *how to use it?* *how to interact with it?*, so that interpretation dispense the maximum user intervention. For that, this ontology is structured into three sub-ontologies: *Profile* (which it does), *Process Model* (how it works), and *Grounding* (map with WSDL) (Figure 3.15)

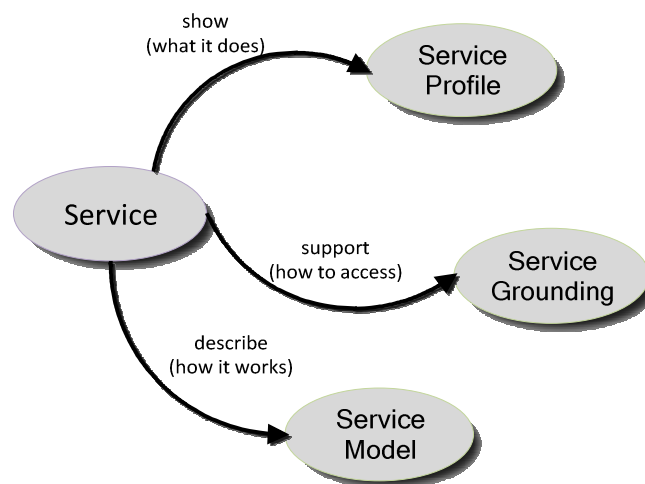


Figure 3.15 - OWL-S sub-ontologies

To be based on processes - *Atomic Process* (AP), defined in an abstract manner (contrary to the WSDL that requires types), which define the *Inputs*, *Outputs*, *Pre-Conditions* and *Effects* (IOPEs),

¹⁹ <http://www.ibm.com/developerworks/library/specification/ws-bpel/>

the OWL-S requires mechanisms to "make the bridge" (*ServiceGrounding*) with the inputs and outputs of the WSDL.

The following sequence of images (Figure 3.17, Figure 3.18 Figure 3.19) shows a simple ontology OWL-S to a particular webservice that manage Hotels, developed with the support of OWL-S Editor²⁰.

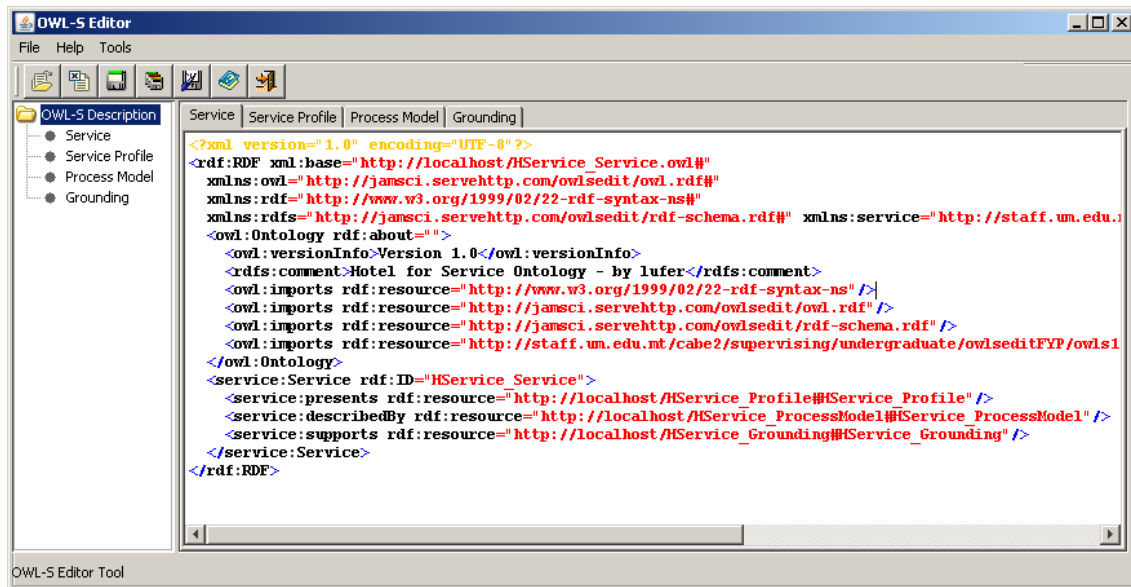


Figure 3.16 - Ontology: ServiceDescription

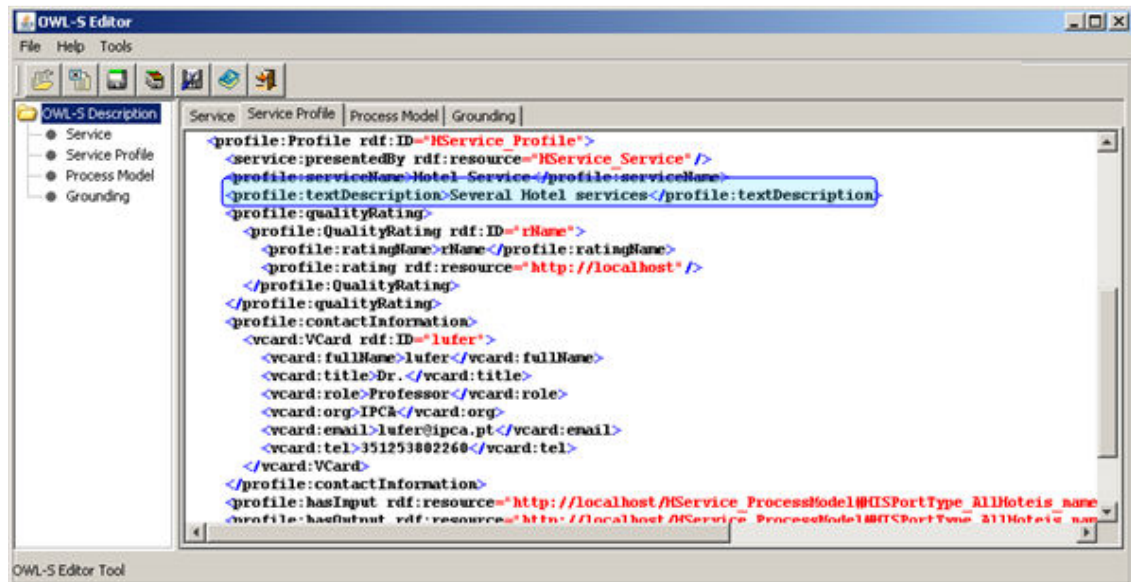
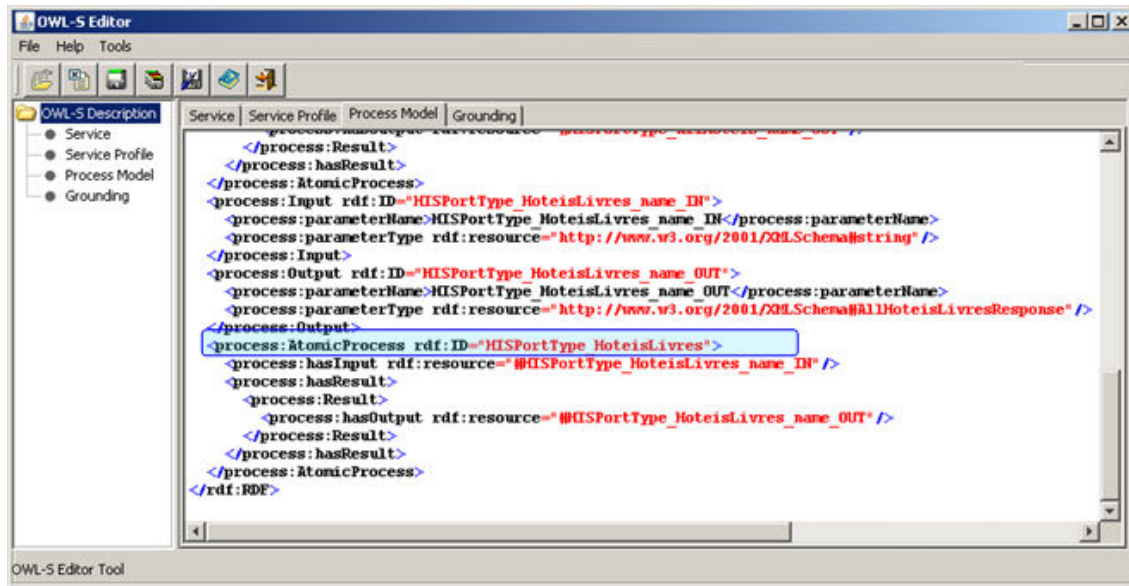
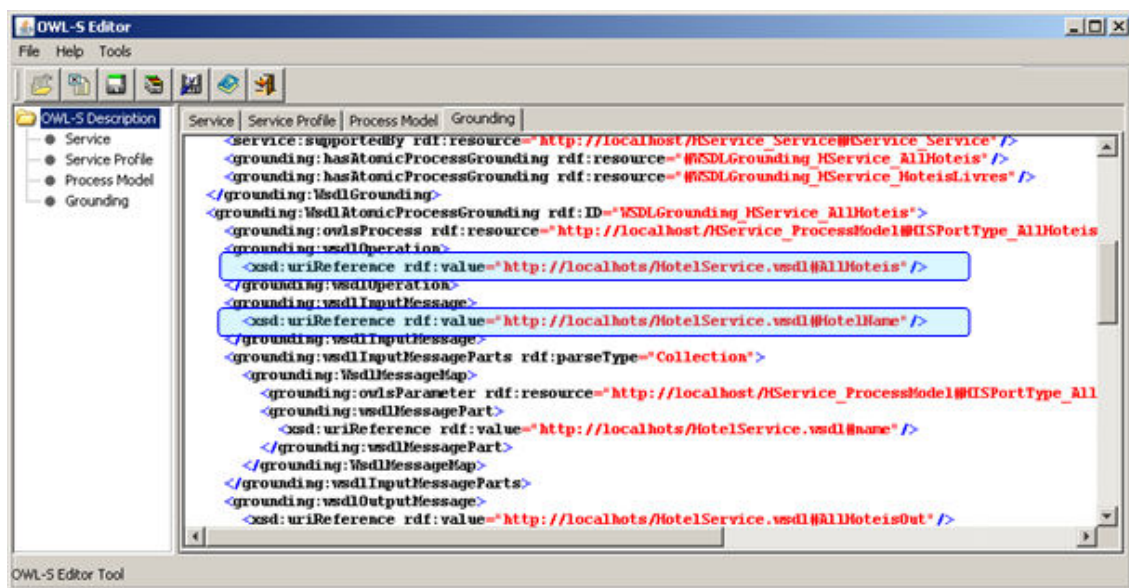


Figure 3.17 - Ontology: ServiceProfile and author's Vcard

²⁰ <http://www.mindswap.org/2004/owl-s/services.shtml>

Figure 3.18 - Ontology: *HoteisLivres* ServiceModelFigure 3.19 - Ontology: *ServiceGrounding*

The *Grounding* of this AP (Figure 3.19) maps to WSDL, when associates the process described to a particular operation, and each element of IO to each part of the messages available (*AllHoteis*, *HotelName*).

Pre-Conditions (invariants to satisfy by who intend to use the service) and *Effects* (invariants to check after the execution of the service) were not condiered in this example.

Furtermore, (Davies, Studer, & Warren, 2006) explored that web services discovery can be improved with semantics in meta-information over UDDI²¹,

The following table summarizes the technology spectrum after the inclusion of semantics in Web services.

<i>Web</i>	URI	HTML	HTTP
<i>Web Services</i>	UDDI	WSDL	SOAP
<i>Semantic Web Services</i>	OWL-S /WSMO		

Table 3.2 - Semantic Web Services Technologies

Remarks

Although it is evident the more capital gains that can be achieved with the addition of semantics in Web services, automating its discovery and composition will only be possible with the implementation of new tools and technologies. The OWL-S has some limitations and the emerging *Web Service Modeling Ontology WSMO*²² already comes with new features for modeling and discovery tools. Semantically it is clear what needs to be done. In practice there are still some steps.

3.8.3 WCF

In a context of multiple inheritances, even in the most recent technologies of Web Services, the demand placed to the programmer and Systems Architect is very high. Although it seems everything simplified, integrating different paradigms, different protocols, becomes a burden too heavy. If we add *J2EE*, *.NET Remoting*, *WSE*, *Enterprise Services*, *Microsoft Message Queue*, *JMS*, etc. is not difficult to reach this conclusion.

With Web services applications are geared to services, are scalable, platform-independent, interoperable and easy to evolve. But they keep a major handicap: they are *state-less* (not persistent).

²¹ can be automated on semantics using for example the WSMOX (<http://www.wsmx.org/>)

²² <http://www.wsmo.org/>

MSMQ, *JMS*, etc., promote the ability of the messages, the *Enterprise Services* promote transactions, security, etc., the *.NET Remoting* offers the automatic generation of proxy and easeness to interconnect different objects, etc. And we could continue to present plenty of technologies that, as is the prerogative of many companies, promoting new buzzwords, terminology, abbreviations, etc., but in essence they are more versions of versions.

Thus, the big note that one should take away from this whole panoply of technologies is that many of them, even if they are from the same company, are not easily "compatible". It is not simple to get one of the technologies, using the potential of others.

Microsoft is a serious example of this scenario, so surprises us now with the development of *WCF* - *Windows Communication Foundation*, to take care mainly with the strategy of the project without "losing" with details of implementation. The integration capabilities are immense, and access to services shall be technologically less conditioning (Bahree, Cicoria, Mulder, Pathak, & Peiris, 2007).

But in short, this new "architecture" is justified, because it comes to allow:

- *The interconnected applications development*: according to the requirements of the existing applications;
- *A unified programming model*: the programmer can abstract away the complexity of any particular technology
- *All the technology used is easily integrable*;
- *Greater interoperability*;
- *Abstraction of infrastructure*: the programmer develops the application apart completely the support infrastructure, i.e., focuses on the logic of problem or desired service;
- *More possibilities of use an external service*.

WCF also took advantage of the "good things" that has inherited from current technologies, particularly from web services. So, as we saw during the presentation of Web services, the services continue to be:

- Autonomous
- With clearly defined and explicit operational limits

- Share contracts and schemas (XSD) and not types or classes
- The compatibility between them follows well-defined rules

The architecture that governs them (SOA), WCF maintains loyalty to:

- The applications are unaware of the changes for services (flexibility guaranteed)
- The location of the services is not difficult (Ubiquity)
- The services are independent of the protocols and formats (Messaging flexibility)
- The services are independent of platforms and implementation (respect the contracts)

However, the technical approach of the WCF architecture, also for having been released more recently, distances itself somewhat to be based on three concepts: the *Service Location* (*A - Addressing*), the *Service Access* (*B - Binding*), and the *Contract* (*C - Contract*) between them. The so-called *ABC of Windows Communication Foundation* (C. Microsoft, 2006).

In short, a process via WCF must respect the following order of events

- Defines the interface of a service and implements through a contract;
- Choose the form of connection to this service
- Installs itself ("publish") this service on a repository, from where it can be used.

From the perspective of a client (user), he should:

- To know where to find the services (address)
- What protocol should follow in order to use these services (binding)

Thinking in SOA, a service is here represented by an *EndPoint* (ABC). It could have multiple *Endpoints* with differences at *Address*, *Binding* or *Contract*. This increases the interoperability of the same service for any level of technology/application.

To structure all these processes were hierarquized a set of objects, being headed by the *ServiceDescription* (for passive entity to produce services) and *ChannelDescription* (for passive entity of the use).

Briefly, a service can contain multiple *ServiceEndPoint* while a client can contain only one. A *ServiceEndPoint* corresponds to a SOA WebService with the ABC entities according defined.

Remarks

So, we should see the WCF as another initiative to try to achieve three important objectives of practically oriented all previous initiatives:

- Improve the interoperability between platforms
- Unifying existing technology, especially dedicated to distributed systems
- Promote the development of service-oriented applications.

3.8.4 Interoperability on Cloud

3.8.4.1 Cloud Impact

New technologies promote new opportunities and new challenges. Usually new paradigms arrive from (considered) deprecated ones since their supporting sciences can take advantage no more on new technology potentialities. New patterns (for analysis, development, etc.) are accepted when previous experiences grant no more efficiency and appropriateness. New features appear when new applications and technical devices prove user's acceptability. All these facts sustain a skewed and restricted purely technical perspective, where tangible measures are easily managed.

But nowadays social concerns are present in almost all kind of activities, demanding human behavior support over the traditional technical-centric submission. The persons need and demand to be themselves and not represented by a machine. This fact sustains the complementary perspective where tangible measures are replaced for intangible and not-supported measures by technical solutions (Goldstein, Lazaris, & Weyl, 2011; Moffitt, 2010).

As answer to this and because of its relevance, new technologies offer new mechanisms or process to allow human approach and consequent more natural interaction and cooperation. A necessary alignment between people and technology.

With the advent of the "clouds" the weather returns again to be unpredictable and the forecast evidences that there are not only good conditions for technological (and commercial) growth.

Coming as consequence of the increased capacity of connectivity, availability of data and “services” and their emergent potentiality, the essence of this new “style” is essentially based on technical infra-structures (Reese, 2009). The announced commercial success mainly comes from its capability to support any infra-structure requirements. Paraphrasing the Expert Group (Group, 2010a), *“Clouds are of particular commercial interest not only with the growing tendency to outsource IT so as to reduce management overhead and to extend existing, limited IT infrastructures, but even more importantly, they reduce the entrance barrier for new service providers to offer their respective capabilities to a wide market with a minimum of entry costs and infrastructure requirements – in fact, the special capabilities of cloud infrastructures allow providers to experiment with novel service types whilst reducing the risk of wasting resources”*. Clearly a technical perspective!

But this new “paradigm” will quickly represent no more than a new technical change if the participants (users, applications and processes) don’t change themselves according to it, mainly in order how they explore their potentiality. For instance the facebook, an evidence of cloud importance, appeared as a social network platform with a lot of new interaction ways (likes, wall, etc.) and tools (chat, video, avatars, etc.) between participants. According to several literatures, its exponential grown is sustainable by a new set of ways and tools which allows their participants to interact, being the *communication* and *availability* the main arguments for this tremendous phenomenon. Until now, only internet (nowadays called internet of things) can be proud for similar happening. In that case, however, due to the excessive “distance” between participants (surfers, managers, producers, promoters, etc.) and consequent lack of confidence between them, has made it nothing more than a data repository. That’s one of the reasons why several internet activities (eCommerce, eBusiness, etc.) took time to grow, below expectations!

It is clear the technical capability and potentiality of the cloud concerning scalability, robustness, security, interoperability, flexibility and many others attributes (Betts et al., 2010). However the emergent cloud services make us feel some obdurate about its effectiveness and knowledge constructiveness capability.

Considering this we proposed an integrated solution for manufacturing support where integration parts are represented by technology and human. Summarizing, cloud technology will support integration technologies and semiotic tools will support human-to-human interaction. The system will be a way to allow humans to (inter)communicate and relate and since co-create decisions.

Since *Cloud Ubiquitous Manufacturing* demands information systems that ensure sufficient ubiquity and availability, and even known that connectivity is not yet ubiquitous, we intend to enrich this technological perspective with social perspective where the user can interact easily with the system and naturally with others users.

3.8.4.2 Real-Time Supporting Technologies

The context where multiple autonomous resources need to collaborate, demands mechanisms that allow their efficient synchronization. This efficiency means the resource capacity to know what is going on with others related resource.

The intended resource requirements to be agile and easily integrated made this synchronization better supported with real-time communication supporting technology. Avoiding to make a deep analysis on technological details and considering cloud based services (resources) where ubiquity is the core key, we center real-time communication technologies on *Threads*, *WebRTC*, *SignalR* and *XMPP*.

Indeed, *Threads* is a not a real-time communication but a concurrent supporting technology pattern that could be significant on distributed and parallel processing (Titus, 2004). So its relevance and potential arrive when using it in some cloud models where distributed data (similar to that existent in each resource) needs to be (asynchronous and in parallel mode) known for any other interested cloud (network) members. The *Task Parallel Library* (TPL) and its *ThreadPool* represents an advanced on *Threads* and offers new potential on parallelism and concurrente applications.

The *Extensible Messaging and Presence Protocol* (XMPP) is a communication protocol for message-oriented middleware. “(...) is an application profile of the Extensible Markup Language (XML) that enables the near-real-time exchange of structured yet extensible data between any two or more network entities (...)” (Saint-Andre, 2011). “(...) where other protocols pull data, XMPP pushes it, allowing more efficient notification and faster responses to new information. XMPP has native support for social features found in many of today’s most popular applications, making it easy for developers to add and build upon people’s relationships and communication.” (Moffitt, 2010). *Google Talk Service* is one of the most common of those applications. However, its use in the implementation of new applications presents some complexity which operates at *Presentation*

Layer mainly, using JavaScript or JQuery. *XMPP Jingle*, *XMPP BOSH*²³ and *XMPP PubSub* are examples of existing and well explored libraries for communication services such as chat, instant messaging, federation protocols and others. Relevant is also, for instance, the possibility of the new generation of VoIP (*Voice over Internet Protocol*) could be supported by XMPP instead of SIP (*Session Initiation Protocol*).

WebRTC, from *Google*, *Mozilla* and *Opera*, is a working in progress real time communication for the web supported by JavaScript APIs. In practice one can admit that WebRTC brings VoIP to the browser natively, even known that all developments are actually done only for browser *Chrome*.

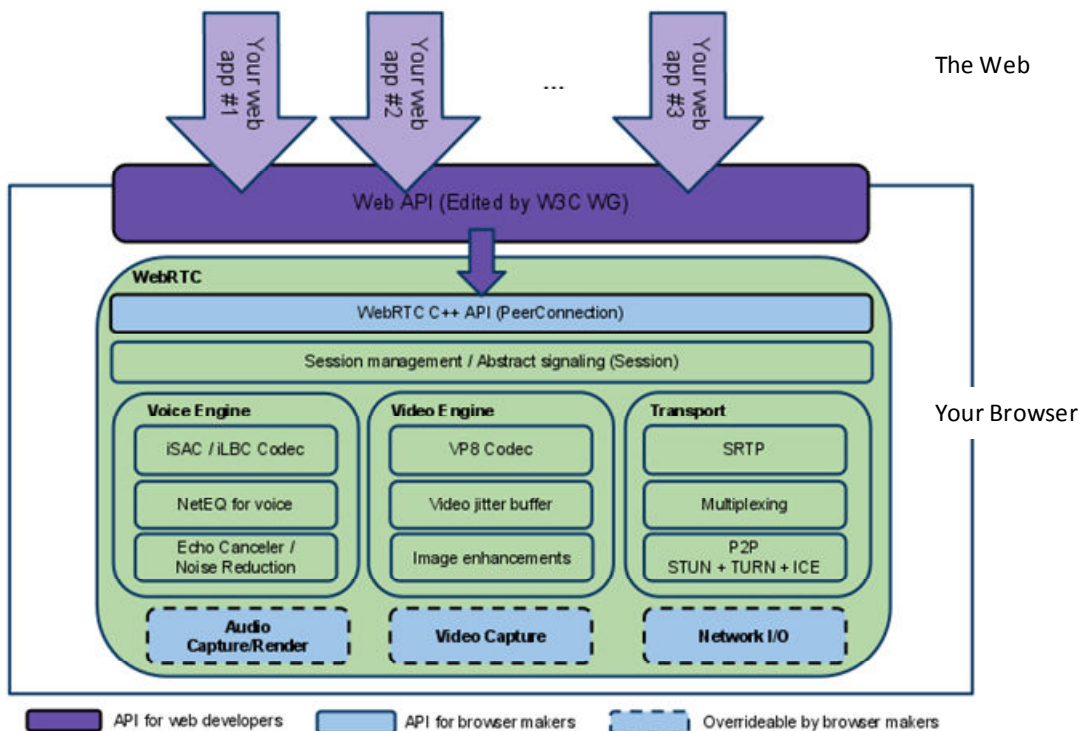


Figure 3.20 - WebRTC Architecture

(<http://www.webrtc.org>)

Relevant on WebRTC is the existence of APIs for browser developers (*WebRTC C++ API*) and web developers (*Web API*) that allows, for one hand, deep browsers exploration and new services development, in the other (Figure 3.20). According to WebRTC organization, “(...) *these APIs should enable building applications that can be run inside a browser, requiring no extra downloads or plugins, that allow communication between parties using audio, video and*

²³ BOSH - *Bidirectional-streams Over Synchronous HTTP* - transport protocol that emulates the semantics of a long-lived, bidirectional TCP connection between two entities (such as a client and a server) by efficiently using multiple synchronous HTTP request/response pairs without requiring the use of frequent polling or chunked responses (<http://xmpp.org/extensions/xep-0124.html>)

supplementary real-time communication, without having to use intervening servers (unless needed for firewall traversal, or for providing intermediary services) (...)", that means, "(...)enabling rich, high quality, RTC applications to be developed in the browser via simple Javascript APIs and HTML5 (...)".

Internet and its HTTP (the fundamental Internet protocol) was not conceived to support complex forms of interaction. For instance, HTTP has no built-in support for state or even security. The basic architecture is very simple and nowadays represents a constraint: a client makes a request and a web server dispatches it, a pure one-way request/response pattern. How can this protocol support social networking? Deficiently!

So, the emergent requirements for ubiquity demand the web upgrade on their actual pillars: HTTP, HTML and Javascript.

The Client to makes a request needs to establish a pool with the server, and must wait until he gets its response. Thus, this request/response implies a lot of server occupation and compromises its scalability. The AJAX "technology" offers *long pooling* to overcome *pooling* handicaps (HTTP overhead). *"(...) with long polling, the client places the request and the server doesn't reply until it has information to return. The Web client keeps a pending connection that's closed only when some valid response can be returned (...)"*²⁴. But *"(...) to be effective, long polling needs some serious implementation work and advanced multithreaded and parallel programming skills (...)"* (Titus, 2004).

If *long pooling* (Figure 3.21) solved some of the existing problems (persistent pools requires less requests to server), the client still needs to wait for the response, to continue his process. *Server-Sent Events (SSE)* arrived to do better than long pooling. A server can push data to the client application whenever it wants, without the need to make an initial request. Moreover SSE is handled directly by the browser.

Meanwhile *WebSockets API* (Figure 3.22) appeared, providing richer protocol to perform bi-directional and full-duplex communication. With this, SSE lost significance. Real-time in both directions is now possible. However SSE (over HTTP) and WebSockets (over web sockets and not all browsers support it, yet) have advantages and disadvantages, when comparing them.

²⁴ <http://msdn.microsoft.com/en-us/magazine/hh882442.aspx>

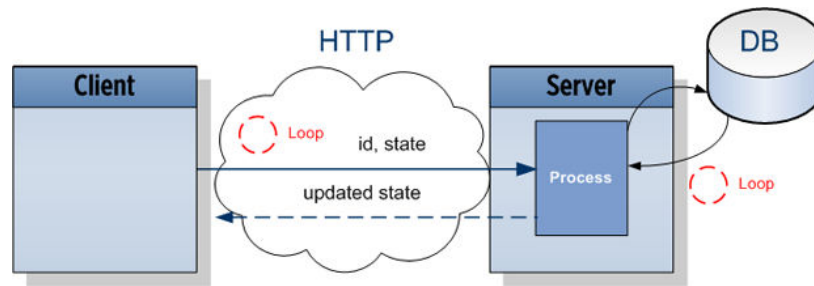


Figure 3.21 - Long Pooling model

(<http://dsheiko.com/weblog/websockets-vs-sse-vs-long-polling>)

SignalR (a library for Microsoft ASP.NET) allow asynchronous scalable web applications with real-time persistent long-running. This brings new real-time support perspectives for many of existing web applications. In practice it allows to stop with request/response pattern and moving again to the one-on-one connection, like in “old times”.

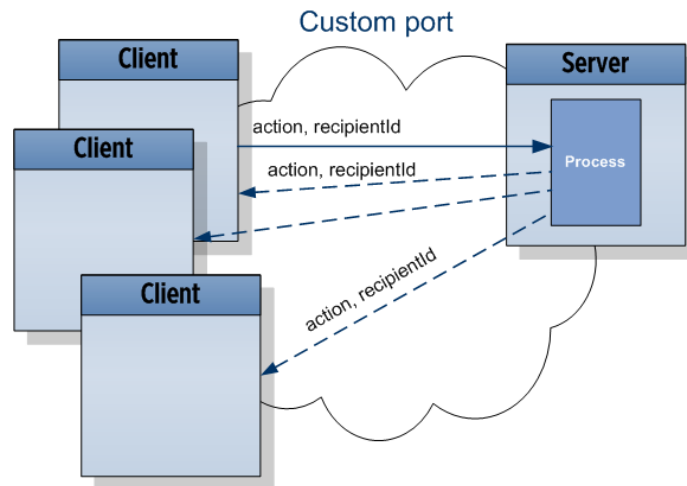


Figure 3.22 - WebSockets model

(<http://dsheiko.com/weblog/websockets-vs-sse-vs-long-polling>)

Thus, on cloud environment where multiple autonomous and heterogeneous components could (must) need to interact, there is a package of technologies that enables efficient interaction mechanisms. The same seems be possible when deal with human-to-human direct communication.

3.9 Interoperability on Virtual Enterprises and Manufacturing

Considering technology perspective, the ability to get interoperability on Virtual Enterprises (VE) is directly related with the information systems that are involved. If this VE are related with

Manufacturing, more issues arrive, mainly those inherent to scenarios of their dynamic reconfiguration, typical on this business activity.

3.9.1 Virtualization in systems integration

There is no doubt that we live in a context in which trust between systems and actors in business processes is based on a virtual image and "benchmark" on previous involvements. What is estimated to pass with the people, in the same way it is reflected in the world around them ... and the business will be one of them (Petty, 2007)

Today reacts to results of stock exchanges, to speculation and commercial transactions, to global requests and betting and/or trends in technologies or processes. Many times companies are going without direction set, pushed by the context of time.

The bet on technologies also suffers this phenomenon of, on the one hand, some requirements virtualization, and on the other, completely virtualization of offering capability. Consider, for example, the CRS Report for U.S.A (Wilson, 2008).

According to a 2006 report from GAO – *United States Government Accountability Office, the Report to Congressional Committees*²⁵, the bet in outsourcing (*OffShoring*²⁶) for software development was at the time one of the six largest investments in all American States.

This represents a changing paradigm if software development and in the bet in software development whose capacity is not possible to measure or predict and whose results cannot be doubtful. It is intended to increase the ability of development and the reduction of costs in this process.

Multivalent and numerous teams erstwhile are now represented by scattered, virtual teams, and whose capacity is sufficient for what you need.

The same is true in business integration and generally in any resource-based integration plan that is not at all possible to control. When these processes involve information systems, then entities involve in exploring process integration technologies, specific infrastructures, models of interaction and collaboration, reorganization plans, semantics, protocols and agreements, all in

²⁵ <http://www.gao.gov/new.items/d06342.pdf>

²⁶ <http://www.offshoring.com/index.html>

order to get the finished product, what now matters of fact, being guaranteed to be provided (G. D. Putnik & Cruz-Cunha, 2005a).

If we focus now in the context of the technologies that we have been addressing, particularly in Web services, we have seen that, as a result of the limited results inherent in existing architectures, it requires greater abstraction, both to enrich their semantic as to increase its flexibility and ability to distributed processing. The virtualization can be seen as an important step to achieve this.

Let's say that this concept is the one that best represents the current trend of integration technologies. As it is possible to virtualize distributed physical capacity (*Grid Computing*, for example), operating systems, displays, etc., we naturally go towards applications virtualization.

In the following Web blog we read:

30 Sep 2008 04:24 PM EDT

Digging Down into Application Virtualization

XenApp enables IT organizations to reduce the costs of delivering applications by centralizing management, security and control of apps and data. Application virtualization technology provides a flexible application delivery system that can select the best method to deliver an application dynamically, based on the user, application and network.

<http://community.citrix.com/blogs/tag/application%20virtualization>

As it is evident in this divulgation, the desired step of nowadays is clearly to respond to the user, not with more technologies that he needs to assimilate and integrate, but rather with the need applications.

Application virtualization requires clear entities (*Agents*) that are responsible for monitoring the evolution of the processes, which will of course also be virtual, the performance and quality of services provided.

But the virtualization achieved, for example, with *Grid computing*, *Clusters* or even *P2P*²⁷, although present enormous computational advantages, do not fit at all, to support the current business solutions, since their business processes are not modulated for this kind of technical solutions.

²⁷ <http://www.gridcomputing.com/>

To alleviate this barrier that separates the real processes from the technologies that implement them, SOA architecture ensures the materialization of these processes in applications and software, offering standards in integration and development of different modules.

After a recent study of *PushToTest*²⁸, where it was found the cost savings achieved by the companies IBM, Oracle, BEA and TIBCO, obtained with the application of a proposed service composition (*Composition Approach*) in the development of large-scale SOA applications, it was noted that such a reduction was due to *Service Virtualization*, creating a layer of abstraction between the use of the service by the customer and the whole process of its invocation. The code produced was much smaller and the reuse and flexibility have increased. This study may be accompanied in (F. Cohen, 2008).

However the application virtualization process will be much more delicate and difficult than achieved with computation. First the processes must transform, clearly.

A case of this type of process transformation is explored in (Ferreira & Putnik, 2008), when promoting reconfigurable services solutions for tourist, in response to demand profiles and contextual constraints.

The tourist will have to leave their traditional patterns towards a complete service offering, with integration of various small services, provided by various companies (travel, accommodation, tourism activities, etc.).

A computer solution can support this type of service only if the interoperability between the different entities that seek to provide services is ensured. For the tourist, any rearrangement or reconfiguration of the service is transparent (virtual). He intends to spend a relaxing holidays.

3.9.2 Dynamic Reconfiguration

ICT are assumedly the essential support of all current business processes. The supporting tools of repetitive tasks (spreadsheets, word processors, etc.), team management (groupware solutions), enterprise applications (ERP, CRM, etc.), business networks (intranets, extranets), web commercial relations (eBusiness, eCommerce, etc.), are examples of that.

²⁸ <http://www.pushtotest.com>

Either by the socialization of the Web (Web 2.0, social networks, etc.), by the inter-enterprise networks, or by daily life of the people, the ubiquity of ICTs and the institution of a genuine knowledge society is seen as inevitable.

As every enterprise has its information systems, tailored to their own needs, their association to cooperate will result only if their systems are able to somehow interact. Thus, the main aim of VE focuses on flexibility and/or agility of the enterprises, already labeled as ubiquitous in nowadays, supported by the establishment of serious relationships (personal, commercial, scientific, etc.) but short and objective to attack specific market opportunity, which tends to be short. Traditionally companies have solid and lasting relations, sustained by the good results of previous experiences.

The integration of different information systems belonging to "members" temporarily attached to a virtual company, needs to be supported by an architecture that enables its agile reconfiguration, i.e., the replacement (or interaction) of a system by (with) other or even integration of new systems, with minimal impact, interaction usually based on the sharing of data (structured or unstructured).

Technically the ability of VEs to interact and collaborate represents the ability of their various systems to be able to exchange and use information to fulfill VE integrated objectives.

Katzy (2003) explored the most relevant at the time was going around the virtual enterprises.

He noted the reorientation and consequent approximation of Concurrent Engineering for issues more related to knowledge management, organizational and integration, and identified three topologies or network models of VE, seen as networks of "partners": a first one, *Supply Chain*, process-oriented and based on long-standing relationships; a second, *Hub and Spoke*, structured in a central company (hub) which coordinated relations with the other members of the network, and a third, *Peer-to-Peer*, based on professional networks and project oriented (Figure 3.23). These topologies conditioned the organizational structure and how it was processed (flow of information, materials, etc.) but did not provide rules for how information systems could interact

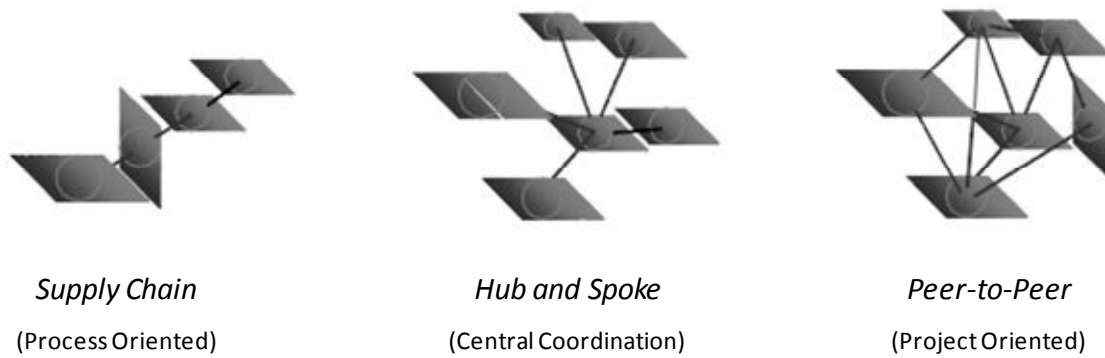


Figure 3.23 - VE Topologies
(Katzy & Loh, 2003)

On the other hand, Katzy found that the most successful VE were supported on solid (usually long) commercial relations (region, product, personal, etc.) which oppose the defenders of *loosely coupled* relations as a means to agility; It is clearly noticed difficulties in the management (or administration) of all parts involved in VE, and emerged the importance of a mediator, a *broker*.

At the time the quick (re)configuration of a VE was not sufficiently studied nor sufficiently sensitized. The entry or exit of new members in the VE was so unconnected to the structure that supported it. The rules of change at the time were based on the types of projects or even in partners and in relations between them.

In the management of VE emerged the first *brokers, business architects, integrators, project managers*, etc., but all of them based on the normal management of enterprises (traditional) and did not have adequate tools to the challenges of the VE, such as support to the distribution of knowledge, decision, etc.

The Virtual enterprises integrated themselves well in an live ecosystem and as such needed to assimilate a set of rules. As most ecosystems have temporary predominantly features, their sustainability results from the community created by living organisms (biotic - enterprises, partners) and factors non-living (abiotic - resources, energy, strategies, etc.) with which they interact and inhabit (Campbell, Reece, Taylor, Simon, & Dickey, 2009).

(Zhuk, 2004) is a study, more in terms of supporting technologies, that demonstrate how the processes, architectures and integration models were being created, chosen or abandoned along the short but complex history of systems integration. If we understand the systems integration (we refer to information systems) as the basis for any other type of integration in a enterprise, the

possibility of its non-integration will represent the handicap so that two or more enterprises are able to integrate properly.

The literature identifies clearly the cause of many failures and that promotes the success of an integration process. One of the most insightful arguments found in the literature was presented by Jhingran (2010), which raises three essential reasons: heterogeneity of data; "federation" and "distribution" of the data; data as patrimony and competitive value of the company.

All of this reinforces the factors presented in (Gazendam, 1999) and (G. D. Putnik & Cruz-Cunha, 2005a), by referring that all this becomes delicate since there are many basic questions on which there is no consensus of opinions:

- If there are different perspectives, concepts or theses on what is really Integrate, the same should happen about what will be integrate VE;
- To what extent is it possible to measure or compare the degree of complexity of any integration?
- The number (in quantity and specificity) of integration solutions continues to grow that difficulties the ability to meet possible relationships between them;

Being VE understood as a new paradigm of organization, existing solutions could hardly respond since they have been created to respond to the previous paradigm. But another variable rises as you think integration achieved. How to measure the quality and sustainability of this integration or? In practice, how to identify the best integrations?

Putnik & Cunha objectively summarize the differences between the two types of integration and prepare two essential theses that summarize all this uncertainty:

- Efficient and effective integration of VE is the main promoter of the own VE;
- For an effective and efficient Integration of VE, it is essential a new paradigm of integration of VE (VEI).

$$VE \leq VEI \leq \text{Paradigma VEI}$$

According to Putnik & Cunha, apart from still need to demonstrate the ability to integrate virtual enterprises, is also imperative to develop a platform that supports the research, development and validation of such solutions, i.e., a *Framework for VEI*.

3.9.3 Ubiquitous Manufacturing

The traditional Manufacturing was superseded. The new dynamic and global business model forced traditional production processes to change in the sense of to be integrated in a global chain of resources and stakeholders. The agility and quick reaction to market changes is essential, and the high availability and capacity to effectively “answer” to requirements is one of the main sustainability criterion. All these performances are considered on Ubiquitous Manufacturing.

But all these, using appropriate information systems, Ubiquitous Manufacturing Systems (UMS) in this case, can only be possible if the efficient interoperability between resources (people, machines, time, services, etc.) is assured. To assured the resources workflow (composition) it requires their efficient integration and mechanisms to coordinate that process. Many of existent infra-structures are already cloud based or are changing towards that virtual architecture. For instance, (G. D. Putnik, 2010; G. D. Putnik & Putnik, 2010a) and (Xu, 2012) suggest a manufacturing version of ubiquitous and cloud computing (respectively) – ubiquitous and cloud manufacturing – and manufacturing with direct adoption of ubiquitous and cloud computing technologies. This manufacturing service-oriented network can stimulate production-oriented to service-oriented manufacturing (Cheng et al., 2010). To use efficiently those infra-structures the applications must be transformed and follows services oriented applications pattern. In this context, resources are seen as services, essentially. Thus, they will depend of services interoperability handicaps above presented, either considering syntactic or semantic (ontologies) interoperability problems.

3.10 Interoperability in Tourism Business

It seems clear that the next generation of e-Tourism infrastructure will have to support flexible automation, integration, computation, storage, and collaboration (Jaaton, Zhao, Rong, & Zhang, 2009). This section introduces some supporting technologies and the latest developments contributing to the creation of global e-tourism solutions.

3.10.1 The Open Tourism Consortium

The emergence of u-commerce, and integration technologies is the backdrop to identifying a series of information products that will improve the searching, management, delivery, and sharing of tourism data. Watson *et al.* (2004) proposed the creation of The Open Tourism Consortium to support the development of several integrated and complementary products, using the open source model.

The *Open Tourism Consortium – OTC*²⁹, is a standby consortium of companies, government agencies, individuals, and universities participating in the open development of publicly available standards and software applications to support tourism activities. Their major goals were to develop a XML based data exchange language for objects and events of interest to tourists (TourML) and an open source parser for this language, able to insert the data into a relational database based on the standard data model. It focuses the capacity to describe touristic information since it could be available in multiple devices. Besides the fact that this initiative promotes u-Commerce and being already supported by a XML Schema, it disables or makes difficult the necessary automatic and agile reconfiguration of a tourism service (Monod, 2004).

3.10.2 Dynamic Tourist Packages: some contributions

Although emergent, the concept of Dynamic Packaging is not specific of tourism activities. Moreover, the concept is not new, having been mainly explored in computer network area, where the *Dynamic Packet Transport* (DPT) protocol proposed an optimized transport protocol suitable to deliver fundamental cost and functionality advantages over existing IP network solutions (CISCO, 2000). Efficient use of bandwidth, multiple-service support, optimization of packets transmission, failure self-healing capabilities, etc. could be some of the features which could inspire software developers and systems architects to adapt the concept to business software applications area.

Considering the current tourism and its computational support, web sites, even being the more common applications in nowadays, are nothing but search tools that offer the tourist some autonomy and new possibilities in defining his vacation schedule.

²⁹ <http://www.opentourism.org>; <http://www.terry.uga.edu/~rwatson/otc/>

Cardoso & Lange (2007) provide a study of the strategic opportunities enabled by dynamic packaging, highlighting the key success factors, stating that an appropriate level of integration of tourism information systems is a key factor for further realizing the strategic opportunities of dynamic packaging. This is consistent with the proposal for tourism supply chain management by Zhang *et al.* (2009).

The Collaborative Travel Agent System (CTAS) based on a scalable, flexible, and intelligent Multi-Agent Information System (MAIS) architecture, is a proposal of Chiu *et al.* (2009) to respond to the increasing demands for ubiquitous access to tourist information systems for service coordination and process integration.

Denicolai *et al.* (2010) explore the relationship between the networking approach of tourism firms and the development of tourism core-competencies, reinforcing the need of solutions based on networking.

The dependence on the context where the activity will take place and the tourist interest and preference (Abbaspour & Samadzadegan, 2008), as well as the application of case-based reasoning and multi criteria decision making on tourism activity planning (Alptekin & Büyüközkan, 2011) are more relevant scientific contributions which refer the main subjects of our research.

3.10.3 Web “Tourism” Services

After the literature review we are convinced that the tourist profile has been changing as well as his interests or preferences, and the emergence of the winning “team” composed by the amazing handheld devices (mobile smart devices) and the ubiquity of the information that anyone can look for (GS1, 2008) is a fact! Despite of the potentiality of these devices, it is not easy for the tourist to plan its tourist activity. This is the actual scenario of tourism in the web!

A new P (from Personalized and Pragmatic) should be put on the previous marketing tourism strategies bet on 8P's Morrison's elements (price, product, place, promotion, people, partnership, package and programming) (Ma & Crestan, 2009), since the tourist perception and interpretation of the context will be important criteria on the final decision.

In another perspective, and due to the generalist behavior of existent web search engines, it is not easy enough for the tourist to find the expected and correct information. However, important scientific contributions are still emerging. E-marketplaces did a relevant effort to specialize these

processes³⁰, The *Travelocity*³¹ service demonstrated the new potentialities of human-computer interaction (Hudge, 2009), Schiaffino in (2009) explored intelligent agent technology to support travel planning, Huang in (Huang & Bian, 2009) reinforced the personalized recommendations systems of tourist attractions, integrating heterogeneous online travel information and advanced selection and matching algorithms (Bayesian Networks and Analytic Hierarchy Processes); Alptekin (Alptekin & Büyüközkan, 2011) integrated case-based reasoning processes and multi criteria decision making (another Analytic Hierarchy Process) system to enhance efficiency in tourism destination planning. Context-based adaptation (Höpken et al., 2008) and context-aware services (Abbaspour & Samadzadegan, 2008), are others contributions which evidence the emergent aware with the context of the activity.

After the emergent technological potentialities observation and tourist requirements analysis, we can conclude that tourism is clearly an activity which claims for services virtualization. A common travel agent will be efficient if he is able to offer services packages geographically distributed. He should have predictable and guaranteed quality of the service, but to archive this, he must to be able analyze the historical quality of services. Having this, it is no longer necessary to sub-contract many enterprises or to physically visit several places to make sure that everything is properly planned. As “essential”, everything must be integrated.

3.10.4 Open Tourism Initiative

The Open Tourism Initiative (OTI) is a semantic architecture (Ferreira & Putnik, 2008; Ferreira, Putnik, & Cruz-Cunha, 2010) able to support the integration and processing of information and/or of processes, disperse and global, of every sort of information that can be (or will be) directly or indirectly related to tourism activities. It describes the structure of Tourism Objects (TO) through specific meta-information and presents mechanisms for brokering them in a global network of those kind of objects.

According to (Ferreira & Putnik, 2008), the OTI “*works like a support layer to grant interoperability between tourism services providers, organized under the format of a virtual enterprise and the support to its subsequent reconfigurations, traduced by the several instances the virtual enterprise suffers along its life-cycle*”.

³⁰ <http://www.e-businessguide.gov.au/improving/e-marketplaces>

³¹ <http://nycgo.com/>

This architecture already focused reconfiguration services, ontologies and brokering mechanisms. In further developments (Ferreira, Putnik, Cruz-Cunha, & Putnik, 2012) complemented the initial architecture with pragmatics components.

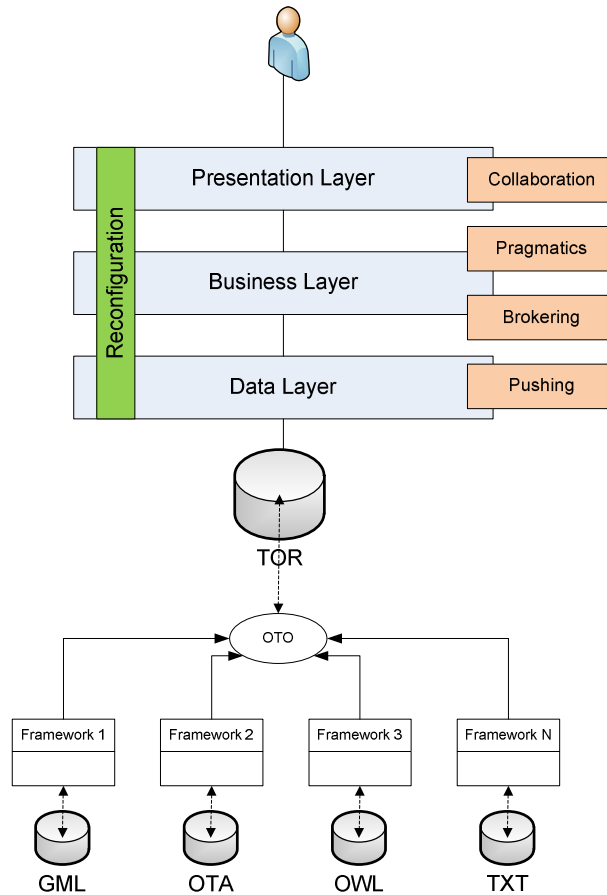


Figure 3.24 - OTI Architecture

(Ferreira, Putnik, Cruz-Cunha, & Putnik, 2012)

3.11 Reflection and Conclusions

It is important to properly contextualize the analyzed integration scenario. It breaks down essentially into three types: *Integration of Information Systems*, *Business Integration*, and *Integration of Virtual Enterprises*. Each one of them is based on paradigms and models with some level of sustainability and accreditation. Risking in assume the integration of information systems as the most accredited, there is not yet a universally accepted solution of integration at this level, indeed.

But if the integration of information systems and the consequent integration of enterprises that use them is strongly shattered as the technologies are evolving, what about a scenario of virtual

enterprises integration where the integration's support infrastructure is difficult to properly characterize or/and sizing, and, seen in a different perspective, reconfigures itself easily and not predictable.

In practice constitutes a form of integration that must react and assimilate new contexts. Putnik called it the *Generative Integration* (G. D. Putnik & Putnik, 2010b).

Although it has not been discussed in detail how, in fact, the development is done, what are the methods used, the technologies used to implement, with more or less rigour, with teams more or less adaptable, via *Continuing Integration*³², *Agile Development*³³, *Extreme Programming (XP)*³⁴, *Formal Methods*³⁵, *Multi-Agents*³⁶ or any other "paradigm" of development and coordination, the fact is that, the quality of the integration between systems is essential.

Pratically all the initiatives advance correcting or supplementing their predecessors. In the current context where distributed applications are essentially web based, the owners of the existent protocols (security, communications, data, etc.), have now the need to "convert" them self, in the sense they can continue to offer the same or new services.

Continues in pattern *Server* (that provides the services) and *Client* (that uses them), they can increasingly be isolated or less need to know each other, to be able to pursue something. To get this they establish a kind of confidence (trust) (under new protocols, contracts, etc.) which guarantees the quality of services among them.

Without entering into sensitive issues such as security, persistence, availability, etc., note that everything unfolds the image of human social behavior. But intelligibility that characterizes us is not yet reflected in applications and we believe this will be the next step.

It becomes necessary some decision-making ability in selecting services and even in the evaluation and reaction of the quality of the services provided.

From the initial aim of sharing documents via *HyperText*, the web jumped so suddenly, reacts now to its socialization (Web 2.0) and prepares to enter on the reality of the demand for

³² <http://martinfowler.com/articles/continuousIntegration.html>

³³ <http://www.agilealliance.org/>

³⁴ <http://www.xprogramming.com/>

³⁵ <http://www.fmeurope.org/>

³⁶ http://en.wikipedia.org/wiki/Multi-agent_system

intelligence. John Markoff (in 2006)³⁷ and Nova Spivack (in 2007)³⁸ have already risked with a Web 3.0 for the next decade. The man wants to take the common sense to all of this, making it in his own image. But maybe it shouldn't be so ... but we believe that always will be!

It was this thinking that led us to explore further in this work the Semantic Web itself that, unanimously, will dominate the fundamentals of integration technologies in future developments.

After a short ride by a long but recent history in diversity and events, we seek to demonstrate the cyclical scheme of events associated with progress and strategies. Distributed computing will have its new cycle, because the new CPUs are now with multiple processors. The present applications must be transformed to better use that potential.

The business processes are governed by the demands of technologies, in particular of information that they must have. The web marked the beginning of an era that regularly back to previous states but with some deviations that make it more efficient temporally. In the beginning web was efficient, today it isn't!

The information systems must adapt to all of these changes, necessarily. Since stand-alone Client/Server applications on desktops, to web applications and now the services-oriented applications, are evidences of that. But this sequence of technologies requires also a sequence in the redefinition of strategies for the business processes of enterprises. Initially everything was made everything "inside" and now "inside just little and acquires a lot".

The confidence jumped of the domains ("walls") of enterprises and now resides in an unknown space, whose proof exists only in the service provided.

The supply capacity of an enterprise is based on the composition of the capacities of others that, interoperating, guarantees the desired service. Web services and SOA architecture are evidences of the distribution of this ability.

The published services can be integrated according to rules, in particular operational and "syntactic" (Sprott & Wilkes, 2004). With the inclusion of semantics in these integration processes, the human intervention in the processes of coordination will tend to decrease and will be possible a transparent distributed operational capacity.

³⁷ <http://www.nytimes.com/2006/11/12/business/12web.html?pagewanted=1&ei=5088&en=254d697964cedc62&ex=1320987600>

³⁸ <http://www.kurzweilai.net/meme/frame.html?main=/articles/art0689.html?m%3D3>

Some people say that Web 2.0 will be the harbinger of a global service-oriented architecture, with its *feeds* and *podcasts* to match the *WSDL/BPEL*, with *tags* and *folksonomy* corresponding to *UDDI*, and *mashups* corresponding to the composition of services, and the browser itself to play the role of the ESB.

If we can operationalize this network of services with a tremendous distributed computational capacity, we will have a *Grid-Enabled Web Services* that, begins, indeed, to materialize with the *Cloud*, with the definition of processes like *Personal Brokers* (new generation of *Personal Agents*), able to decide on behalf of man.

But important issues will continue to prevail: how to find the best service, its performance, security and reliability, when is something that is not controlled? And the transactional guarantee?

Rapid change, fierce competition and an ever-flattening world economy are driving the need for superior business agility. A new class of truly agile organizations, the globally integrated enterprise, is emerging as the winner. How? By delivering unique value, tapping into the power of globalization and forging a strategy of componentization.

These organizations understand that using service-oriented architecture (SOA) is a preferred method of delivering sustainable agility. They need this agility - that is, the ability to quickly and effectively respond to changes, opportunities and threats - to effectively compete.

4 SEMIOTIC INTEROPERABILITY FRAMEWORK

This chapter continues to present the context in which fits this research. In essence the work relies on the relation technology/human and in the ability to create mechanisms that allow the two parts interoperate and constitute a whole.

4.1 Pragmatics

Pragmatics is one of the semiotic fields and concerns the relation between ‘signs’ and their interpreters (Morris, 1938). The ‘sign’ is the foundation of semiotic theory, formulated by Saussure (1916) as a ‘dyadic’ model: *signifiant* (the form which the sign takes) and *signifié* (the concept it represents) and by Peirce (1958) as a triadic model: *representamen* (the form which the sign takes), *interpretant* (the sense made by the sign) and *object* (to which the sign refers). Both authors formulated a theory for the relationship between the elements of their models: *signification* (Saussure) and *semiosis* (Peirce) which results in a different argumentation for the same proof: all elements must behave as a whole. Paraphrasing Saussure “*you cannot have a totally meaningless signifier or a completely formless signified*” and Peirce “*nothing is a sign unless it is interpreted as a sign*”.

For example, in linguistic terms, the word ‘full’ (used, for instance, when a recipient cannot have more contents) is a ‘sign’ with: *signifier* (the word ‘full’) and *signified* (the recipient cannot have more), according to Saussure. But the same *signifier* (‘full’) could mean different *signified* and thus be a different ‘sign’ (‘full’ as ‘have no patience’, for instance). Another example, the semaphore’s red light as a ‘sign’ have: red light (the *representamen*), cars stop (the *object*) the idea that the red light indicates that cars must stop (the *interpretant*), according to Peirce. But how it is perceived the same element of those who know nothing about traffic?

Each one of these examples exposes well the meaning of pragmatics because, and paraphrasing Charles Morris (1995), “deals with the origin, uses and effects of signs within the behavior in which they occur”. The fundamental, qualitative, differences between the pragmatics, semantics and syntactic, are virtually the best described by Carnap (1942), based on their degree of abstractness in relation to complete signs and semiosis:

“If in an investigation explicit reference is made to the speaker, or, to put it in more general terms, to the user of language, then we assign it to the field of pragmatics... If we abstract from the use of the language and analyze only the expressions and their designate, we are in the field of semantics. And if finally, we abstract from the designate also and analyze only the relations between the expressions, we are in (logical) syntax.” (Carnap, 1942, p. 9) (cited in (Recanatì, 2004)).

The implication is that any (information) system that aims at considering true needs of a customer, i.e. the needs closest to the real customer's needs, with as less as possible abstractions, should consider pragmatic aspects of communication with him.

Sign interpretations are, thus, context dependent, meaning that actually it is hardly possible to exist an 'absolute', common and universal, interpretation of reality (in our case the reality of the customer needs), but, rather, there are multiple interpretations by multiple communities (i.e. specific for each one customer and by multiple scenarios for satisfying his customer's needs) and in different times (i.e. and in continuous change).

4.1.1 Semiotic Integration: much more than ICT

In the technological perspective the emergence of pragmatic web was a tentative to support pragmatics aspects, complementing the syntactic web (common web) and the semantic web. This initiative tried to get relevant information applying human interaction, i.e., concern not only with the form but also with the meaning of the information. Since pragmatics is a field, rather than a discipline (however, there should not be confused with a discipline of pragmatics when applied within the human communication), and, additionally, belonging to the human communication, the *tentative to implement the pragmatics in an information system as its part is a paradox*.

Other technological initiatives explored several collaborative mechanisms with semiotic frameworks but were no more than technical experimentations to give some intelligence capacity to existing technologies, as happened with agents or web services (Booy, Liu, Qiao, & Guy, 2008; K. Liu, 2008). Once again these attempts tried to “transform” human particularities following to technical requirements towards their integration (utilization) in the information systems.

In nowadays real contexts, the customer's expectation satisfaction must not be seen as an easy and completely defined process. The tourist, for instance, participates as a customer in a set of

complex and unpredictable scenarios where the conditions might be completely unpredictable. In another completely different example, the production manager of a Manufacturing company has to deal also, every time, with unpredictable states (employees health, energy dependence, external factors, many others) even he thought that everything is controlled.

Considering several distinct scenarios we identified three main dimensions of their complex and unpredictable behavior:

- The *linguistic competence* on communication
- The *behavior* of the responsible (tourist, manager, etc.) during context evolution
- The *technological* conditions

Although most of people think that technological problems (legacy systems, not integrated systems, insufficient support, methodologies, etc.) represent the main argument for the deficient alignment among, for instance, tourism business and IT, we are convinced that personal (tourist) factors represent the strongest argument, most of them related with the ability to well communicate (in sense of to be able to transmit and understand a message) or with the dynamic behavior of the tourist, in this case. Let us explore these dimensions better with some possible real and practical scenarios.

Considering the language meaning, a subset of linguistic knowledge³⁹ (Fromkin, 2000), present in the intra-tourist (or agents) communication, several factors (educational, cultural, social, religious, intends, etc.) can easily respond for the high probability of incapacity, error or failure in the meaning transmission process. This means that any two persons in the context of tourism (tourist agent and customer, for instance), might have difficulties in communicating. Paraphrasing Mey, the ability to understand another speaker's intended meaning is called *pragmatic competence* (Mey, 1993). So, have the capacity to communicate cannot be enough.

In a completely different aspect (dimension) of the scenario, the tourist, as human, could easily change his interest or motivation regarding a given objective, depending on the context where the activity is to take place as well as his new interest or preference. The tourist may have had presented their initial requirements; they were well understood for the tourism Agent (so the first scenario was surpassed), and the activity was prepared according to those requirements. But the

³⁹ Language form, meaning and context

tourist can easily change them or have new ones, later on. This is a typical situation where the tourist, independently of any information systems or language problems, changes his behavior or interest. Since the human behavior is not constant (most of the times the behavior is irregular or ambiguous), the patterns of behavior are not more than empirical or just a representation of part of the real information.

In the technological dimension of the problem (and not only informatics) and according to the tourist's requirements, the system will suggest a set of possible activities. In case of doubts or indecisions about what activity to choose, what to do when the activity changes or when his interest diverges, the tourist will need to have more (new or different) information or even to interact with someone (tourist agent, another tourist, etc.) in order to refine some requirements or to clarify eventual (new) questions. A great effort of interoperability among all tourism services providers are the essence for effective tourist support. If those particular systems are not interoperable and somehow integrated, the "global" system hardly satisfies the tourist expectation.

The regularity with which these scenarios can happen requires agility on the management of tourism service composition, of tourist request as well as the capacity to allow tourists to communicate each other and generate their own activities outside the idiosyncrasy of the information systems.

4.1.2 Collaborative behaviours and supporting technologies

Although it may look different, the communication model persists today as the three entities Shannon and Weaver model (1949) and follows its inherent transmission pattern. As in the beginning, it is need a transmitter, a receiver and a channel as the medium used to transmit the content of the message to receiver. With obvious different technical support, the systems continue to be classified as discrete, continuous or mixed and suffer with "noise" problems too. The actual agent (foregoing transmitter or receiver) of the communication use the team (mixed), virtual (continuous) or face-to-face (discrete) models to collaborate (foregoing communicate) and the "noise" resides in things like confidence ("men in the middle" pattern), trustiness, etc. So, if in that time these were technical particularities, now we assume the analogy more to the way how and for what they are used for.

As Weaver defended, the accuracy (technical), the precision (semantic) and the effectiveness continue to be the critical levels of actual communication goals. The syntax (form), the semantics (meaning) and the pragmatics (use) of the language, are the essence of these levels, respectively. The terms syntax, semantics and pragmatics were introduced in linguistic and semiotic theory of Ferdinand de Saussure (1916).

This dynamic collaborative behavior might be further enhanced with the emergent technological opportunities. In nowadays information society the persons are focused on common electronic social media as form of collaborative systems. The people are adopting a new social cyber-behavior, motivating them to adopt new habits in working as well in thinking (Mickel, Agosto, Vignollet, & Marty, 2006). We are now better related persons and we can easily share our point of view or send intended information. However, there is an insufficient utilization of this new capacity in actual information systems. The majority of systems were made to minimize the human dependency in the decision making and reduce the complexity (the human being is naturally complex). In consequence of this, the actual systems are “distant” from human being and can hardly be fully functional to him. Although the user can more easily interact it is difficult (almost impossible) to “pass” his interpretation of the context to the system. The system does not need that information to work too. It is a mechanical behavior.

Paraphrasing Giuseppe Begnis (2010) *“the behavior of the collaborators and the collaborative artifacts are affected by the ability of the infrastructure to facilitate desired and appropriate behaviors”*.

The increase of technological capacities (considering devices and applications) for real-time social interaction, using on-line meetings, distributed multimedia brainstorm, synchronous and virtual interactions, etc., as evident on *facebook, twitter, skype, twiddla, thinkature*, etc., can be models to follow or to integrate on future applications. Since pragmatics is possible when human beings can share and react directly among themselves, if the information systems support it, the information systems will be (more) aligned with user’s interests and improve the result of the collaborative effort.

4.1.3 Towards Human/User Interface alignment

The dichotomy scenario of human context and machine intelligence continues to have strong defenders. (G. D. Putnik & Cruz-Cunha, 2007; G. D. Putnik & Putnik, 2010a) still continue to

place emphasis on that, and TiiS - the ACM Transactions on Interactive Intelligent Systems (TiiS) (Hartmann & Schreiber, 2011; Jameson & Riedl, 2011), alert for the intelligent systems that people will need to interact with.

According to Hartmann (2011), *"...there is an undeniable ongoing trend to put computing capabilities into everyday objects and places...these smart objects are fully functional on their own, but added value is obtained through communication and distributed reasoning. While other venues have focused on the many technical challenges of implementing smart objects, far less research has been done on the topic of how the intelligence situated in these smart objects can be applied to improve their interaction with the users..."*

Interaction with smart objects is situated in the physical environment of the user, i.e., it does not take necessarily place in a desktop setting. A smart object often uses additional cues from its context to improve the interaction with the user, thereby, making the interaction between user and smart object feel more natural. Furthermore, a smart object is a physical object which allows to exploit approaches from tangible and embodied interaction to enhance the interaction".

Emergent technological devices (smartphones, iPads, etc.) already support real-time collaboration. People talk and see each other at any time, in the way they want, and existent applications try to explore those new capabilities as an add value. This new form of interaction and consequent people massive adhesion, promote business models changes. Thus, to better align with human requirements, future applications need to change significantly, to be integrated in new devices and to be seen as the common and essential tool for human life.

Figure 4.1 resume the mains technological steps which happens during this changing process: 1) new applications for new requirements addressed by multiple "isolated" applications; 2) related applications offer more services and thus perform better functionalities; 3) complement with external services (SaaS) allows better strategies and costs reduction; and 4) in-the-cloud services composition allow applications to be closely aligned with business tasks.

The first big step to this change was done with SOA, which aroused developers for the loosely-coupled need of portable and upgradable applications (Mulholland et al., 2008). Now, cloud architectures waked up system architects and software engineers for a new potential of processing, after the success (or complexity/unsuccessful) of grid and cluster computing (Reese, 2009). So, *cloud computing* is now the big subject.

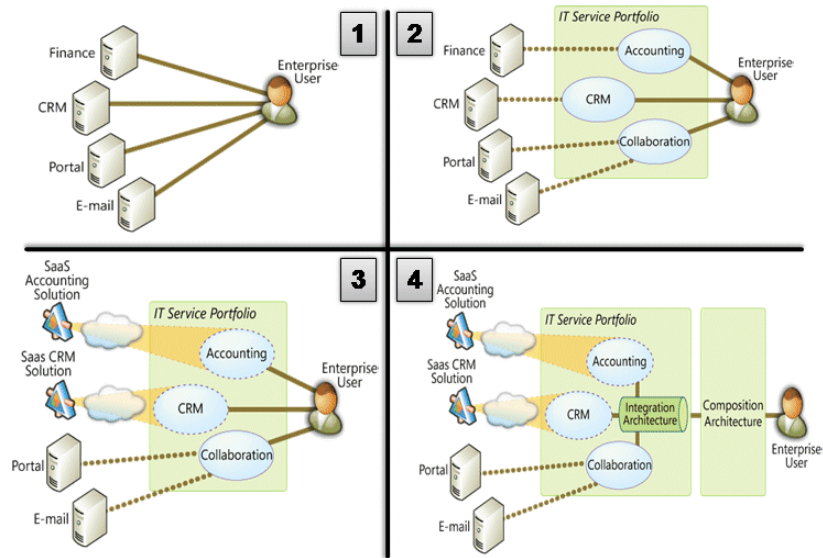


Figure 4.1 - Solutions archetypes are changing
(Carraro & Chong, 2006)

In conclusion, emergent software solutions result from the composition of several services. Those services are efficient and don't require great infrastructures investments or maintenance contracts (Harding, 2011).

This pure technological process and the emergent requirements for services ubiquity and UI multimodality (as previous referred in UMS), transform and enriches applications with real-time communication services support (chats, videoconference, conference rooms, others). However, this new communicational capability will never result in real effective systems (Ferreira, Putnik, Cruz-Cunha, & Putnik, 2012). Paraphrasing (G. D. Putnik *et al.*, 2011) “(...) *the human-to-human synchronous collaboration (video, audio, etc. and related auxiliary tools) which allows the natural involvement of the user on the co-creation/co-design (co-management) processes with other agents (humans) is the responsibility of the Pragmatics (...)*”.

4.1.4 From User Interfaces to User Experience

Tim Berners-Lee (1994) foresaw that, besides the Internet, the future interest will focus on communication capability. The communicational capability of emergent devices causes continuous and dynamic business models transformations. The initial interest to spread enormous quantity of information is overtaken by the possibility to easily contact other person, in nowadays.

Software applications are usually developed aligning user needs to technology capacities, being user needs essentially resultant from business models requirements. This cycle happens in a context where machines are projected as human substitutes. Interpreting Berners-Lee perspective (1994): “(...)this means that machines, as well as operating on the web information, can do real things. For example, a program could search for a house and negotiate transfer of ownership of the house to a new owner(...)”; it is easy to infer that the human is seen as only a user (or observer) and cannot participate on it.

Although previous technological investments essentially cared to offer sophisticated and efficient applications user interfaces (UI), actually it is required more intelligence and human experience participation (Harris, 2008). Since the initial desire to have “everything” (ad hoc) on the web (Web 1.0) and the “yes, I can” position that allowed anyone to easily publish web contents (Web 2.0), next step (Web 3.0) demands intelligence to get the real value of things, where anyone can generate business applying his own user experience (UX).

Questioning Johannes (2010) perspective, are we really walking towards a value centric culture? And if so, is the human part of that value?

4.2 Pragmatics vs Pragmatic Web

The Pragmatic Web appears in order to reinforce the form (syntactic) and meaning (semantic web) of information on the web, in order to make it more useful to whom (person and not machine) actually needs it (Figure 4.2).

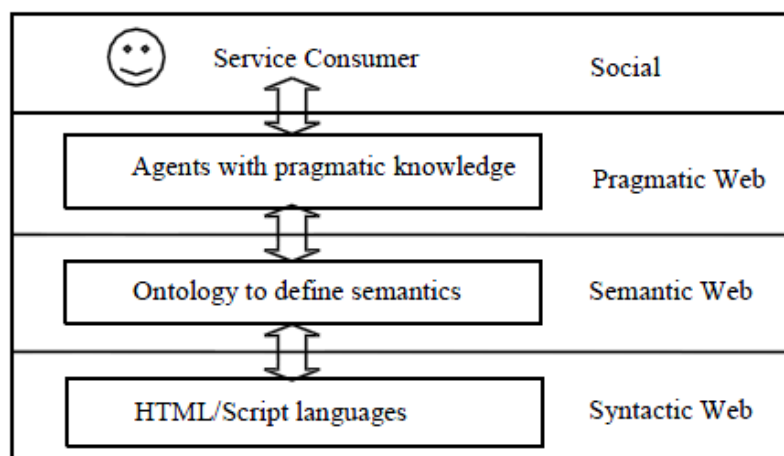


Figure 4.2 - Web Conceptual Model

(K. Liu, 2008)

But the Pragmatic Web fails when plans to develop ways to allow the technology to support this strengthening, with ways to represent acceptance, understanding or disagreement of the people before the concepts. It reinforces the ontologies and even defines new or follows existing pragmatic patterns (Moor, 2005). Attempts at all cost to make more intelligent applications. Once more, only a technological bet.

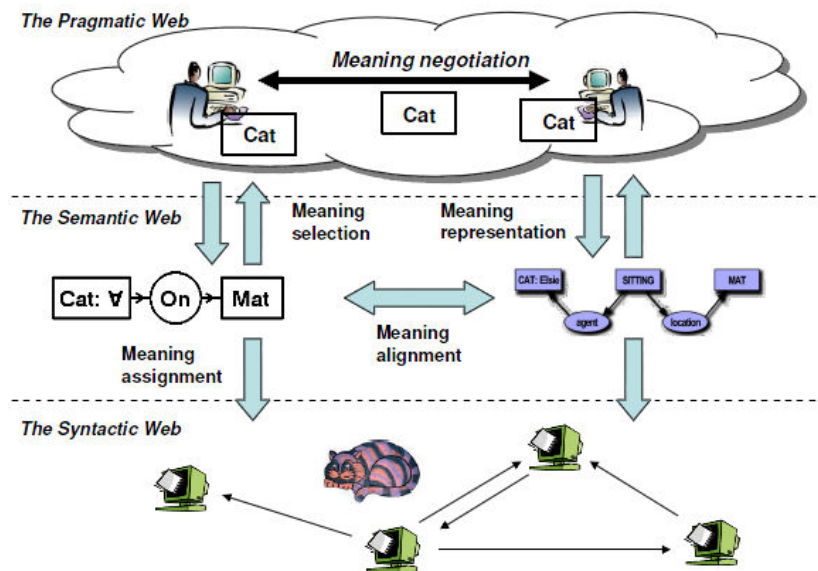


Figure 4.3 - Pragmatic Web as a negotiation tool

(Moor, 2005)

According to Moor (2005) the meaning is essential to “connecto web layers” and that meaning is easily mapping in technological formats (XML). But the key is moreover. Paraphrasing Tamani (2007), “(...)ontologies however, as pragmatic web researchers argue, should not be exploited as fixed conceptualizations of some domains, as they are, but rather as dynamic structures which co-evolve with their communities of use. Members of a community have to communicate and continuously negotiate on their shared background/context. Ontologies must be able to co-evolve(...)”. However, as we previously argue, the tentative to technologically implement the pragmatics is a paradox. Allow persons to participate is essential. The technology is a tool to optimize that and not to substitute them.

4.3 Semiotics in Virtual Enterprises and Manufacturing

This chapter focuses the semiotic framework on our practical research context: Manufacturing and Tourism.

Semiotics on Dynamic Reconfiguration

The ability to integrate virtual enterprises is sustained by Putnik in semiotic framework (G. D. Putnik & Cruz-Cunha, 2007). Putnik argues that the framework for integration of VE arises in levels 4-6 of semiotics (Stamper, 1999) (Table 4.1).

Social Pragmatics Semantic	VEI
Syntactic Empirical	Computer Architectures
Phisical	Technological Infra-structure

Table 4.1 - Semiotic Ladder

As we saw previously when explore the Pragmatic Web and diverting by now a little focus to procedural issues, if we analyze the evolution of the Web (services on the internet), we can deduce that the journey started by offering content governed by syntactic issues, followed, with the emergence of services, the required semantics (T. Berners-Lee, 2001) and now, with the socialization of the Web, the pragmatic nature begins to clear up. Today the value of information is determined by the context in which it is used and by whom.

In a more technological perspective, business success relies strongly on the ability to support business processes flexible but robust and capable of responding efficiently to the rapid and unpredictable demand.

We saw previously too that Web Services/SOA ensures the integration, bu following only technical requirements or sometimes procedural too. However the quality measurement requires other criteria in addition to the "be appropriated and working". It is essential to be able to interpret and decide the best. Full width semantic attributes, the interpretation of Web Services (W3C, 2005) can be (and sometimes is) different, so whom interprete them and the context in which it is done allow it (*pragmatic heterogeity* - (Overhage, 2002)). The context is constantly changing and those involved could be other.

It is in this scenario of intentions and judgments that pragmatics, part of semiotics, when considered, promotes and justifies the emergence of the Pragmatic Web (Schoop, de Moor, & Dietz, May 2006).

By creating agents with this ability to deduce information on the relevance and usefulness in the context, they can be used in resource selection and decision-making, ensuring the cognitive ability in dynamic reconfiguration of virtual enterprise networks and a Human-Machine-Human relation (Figure 4.4) best supported (G. D. Putnik & Putnik, 2010b). This is the foundation that will support the Framework to develop.

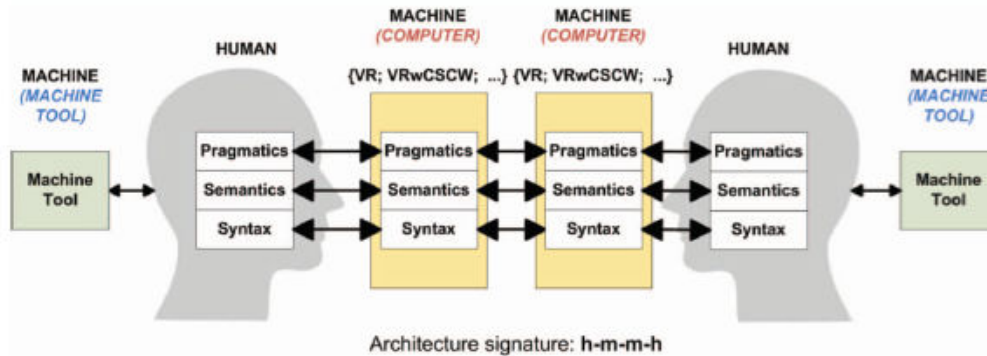


Figure 4.4 - Human-Machine-Human relation

(G. D. Putnik & Putnik, 2010b)

Even talking about Ubiquitous Manufacturing, and considering UMS a kind of dashboard application which allow the monitoring of involved resources, the capability for user (human as resource) to interact (under multiple ways) with other (resources) is essential. If the involved resources are humans, the possibility for them to easily communicate is critical, and, if not, the possibility to immediately knows and react to any unpredictable particular resource detail (capacity, occupation, availability, schedule, etc.) could be crucial. Even technology can offer relevant important information about these (and others) details, all these features are only effectively supported and assured if pragmatics instruments (like conversation, audio recording, video-conference, etc.) are effectively available in the system. Pragmatics instruments sustain a generative integration of users and supports UMS concept (G. D. Putnik & Putnik, 2010a).

4.4 Semiotics in Tourism Business

As we could saw previously, indeed, the Open Tourism Initiative (OTI), with the Tourism Virtual Enterprise (TVE) as the underlying organizational model, could be seen as a set of semiotic-based models in continuous change, i.e. a set of communication models, or a set of pragmatics based collaboration decisions (following the semiotic-based systems integration (G. D. Putnik & Putnik, 2010a)).

Reconfiguration in Tourism is directly related with Tourism Dynamic Packages purpose. After achieved the required technological support it is essential that the tourist can deal naturally with his tourism activity. In an exceptional situation he wants an appropriate solution. If he need to interact to someone related with his activity he must be able to do that.

If everything works around an information system, the same must be possible with the communications and collaborative decisions, towards a co-realignment of his activity to his expectations.

4.5 Reflection

The interoperability is so dynamic that cannot sustain a standard of integration. It surpasses the technological aspect.

One can admit that there is a kind of fidelity to information systems. The user experience is mainly applied to web browsers so web applications need to care well that. But new devices arrive and with them new technological strengths and opportunities arrive too. Persons like to use new things and easily adapt to them. Since all this fidelity is committed, indeed (Figure 4.5).

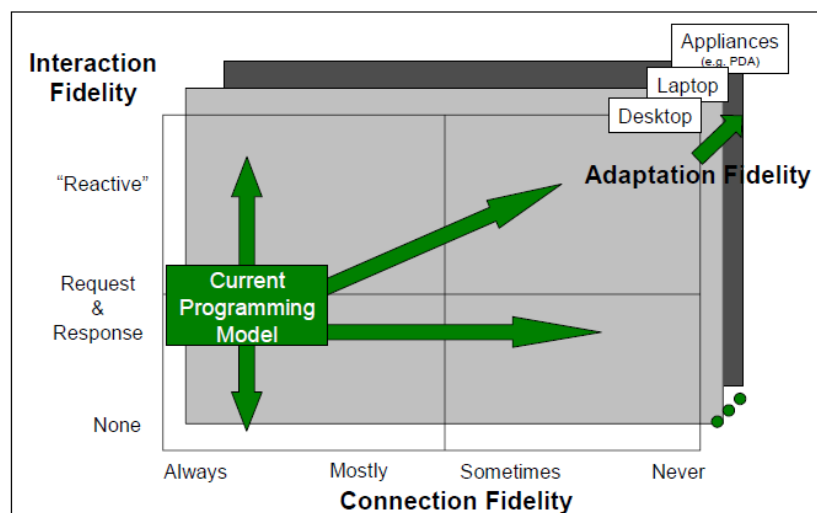


Figure 4.5 - Technology Fidelity

(IBM, 2006)

5 COMMUNICATIONAL ARQUITECTURE MODEL

Several facts show that the more independence from technology, the best architecture robustness and flexibility we get. Since technology does not stop evolving, everything which depends on it, needs to change too. So, how is it possible to stay indifferent to these changes and get the most advance on its potential? Should it be possible with an abstraction layer?

This is the context of our proposal. Any pure technical solution will be limited by its own technology by one hand and functionally limited on the other, because it cannot support all requirements. Many requirements were not initially specified since most of them are not possible to be. There are a lot of requirements which come from the participant's human cognitive capability, meaning tacit knowledge impossible to be modeled in the technical specification. However, it must be considered in the decision taken.

5.1 Transactional vs. Communicational

Avoiding to get into time issues or matters of relevance, it is important however to present what in essence distinguishes a Transactional Architecture for a Communicational Architecture. In a transactional context the emphasis is given to the "state" in which the system is while in communicational the emphasis is given to the ability to communicate or form of interaction with the system.

Objectively, a normal computer application (seen as traditional) bases its behavior on a state evolution, either at the time of its development, and in every moment that it runs (or is used). Several artifacts are used to describe the desired semantics for those transactions. *State Diagrams* (Figure 5.1 (a) and (b)), *Graphs* (ex. *DFA Deterministic Finite Automaton*) (Figure 5.1 (c)) and *Flowcharts* are just some of them. The UML, for example, offers several of these artifacts able of representing this string of states (S. W. Ambler, 2004).

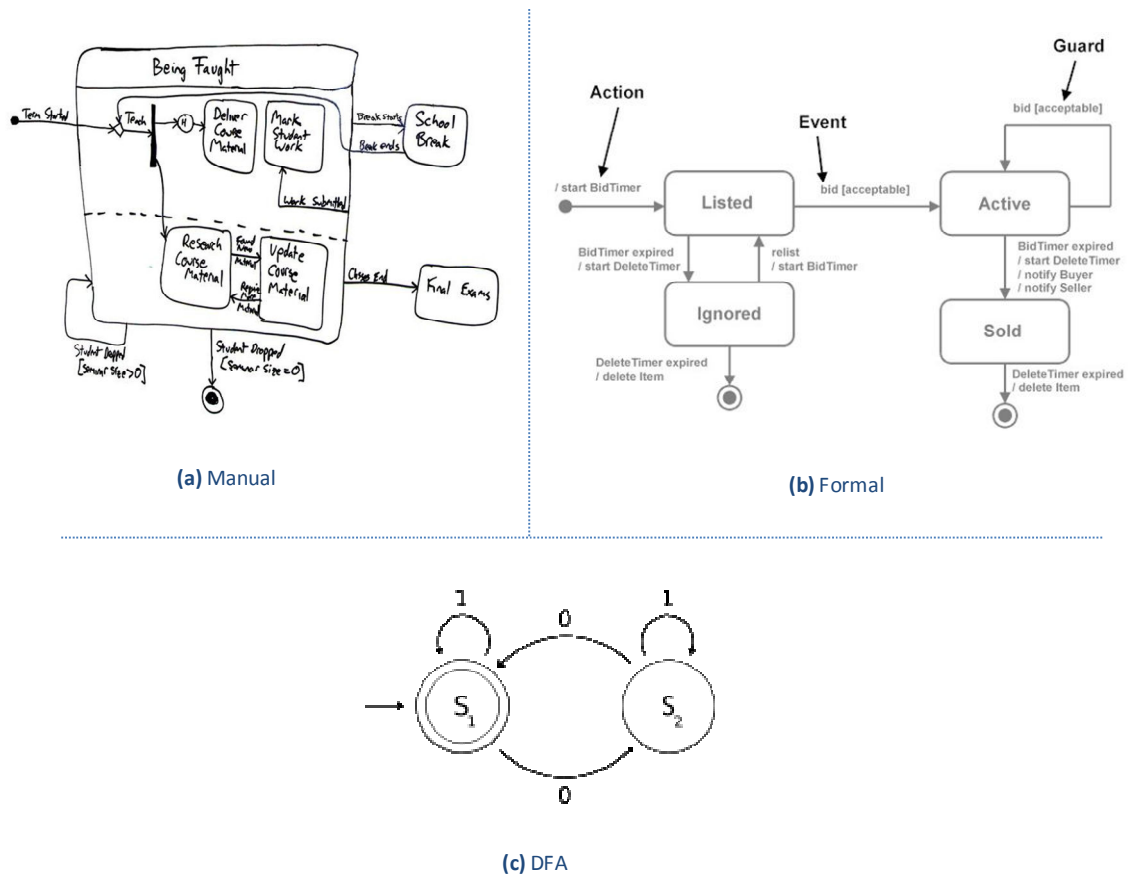


Figure 5.1 - Modeling the transaction of states

The agility with which these artifacts respond to continuous change of requirements is clearly a concern, but it becomes even more critical if this "dynamism" occurs during the use of the application. Several practices of *Agile Modeling (AM)* (*Rational Unified Process (RUP)*, *Enterprise Unified Process (EUP)*, etc.) (S. Ambler, 2003; S. W. Ambler, 2011) seek to respond to this concern, but they can do it essentially at the designing stage, trying to engage all critical parts of the project (customers, in particular) in the process of its analysis and specification.

In order to make these processes the most efficient (and effective) possible they need to use mechanisms that allow all members of the team to communicate with each other, going clearly towards of *Media Richness Theory* (Daft & Lengel, 1986) (Figure 5.2). Factors such as physical distance, temporal proximity, cordiality, etc., affect the ability to work in a group. And even though that exist collaborative modeling tools, virtual or direct, and even accept that Daft's theory emerged as very advanced idea at the time (because at the time the internet did not yet showed its potential), this applies only in the solution design and development phases.

But this theory alone becomes unable in an emergent context where the relationship is increasingly not personal but social and global, i.e., in the other side of the channel there is an entity (resource: person, machine, service, etc.) that is not (necessarily) known but from who we hope to get a reliable service and obtain the feedback at that time (Dennis & Valacich, 1999; Sevinc & D'Ambra, 2004).

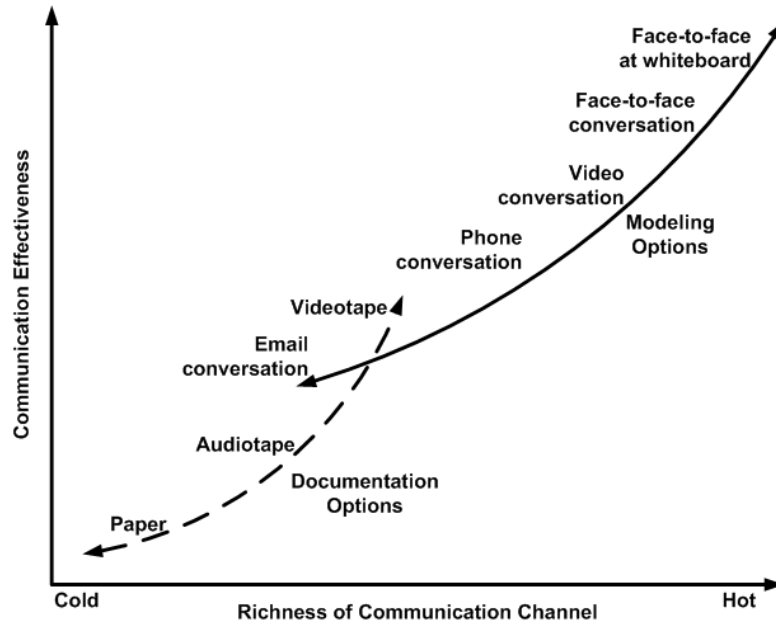


Figure 5.2 - Effectiveness in communication mechanisms (media)

(S. W. Ambler, 2011)

Thus, a communicational architecture manifests itself essential in the application support that, during its use, ensures all necessary and sufficient ways so that any user can interact in a natural (and human) way, not with the system itself, but with another entity (human) present in the system (G. D. Putnik & Putnik, 2010b).

How this should be achieved must also respond to the normal way as a person interacts with other, usually talking, seeing her and talking to her. In technological terms, it is similar to skip from a multi-applications model (complex) to (the simplicity of) a transparently integrated multi-service model. In practice, if we need to talk to someone, we talk! Pragmatically, it is intended to align the solutions with the human behavior, and not only on the abilities to communicate.

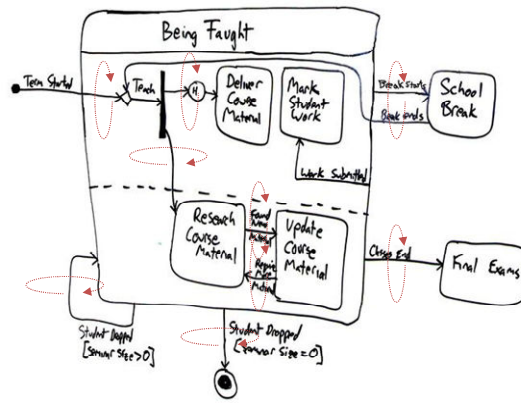


Figure 5.3 - Manual schemas to get effectiveness in development process

In the development process (Figure 3) the human-to-human interaction is possible at any time and with any entity considered necessary.

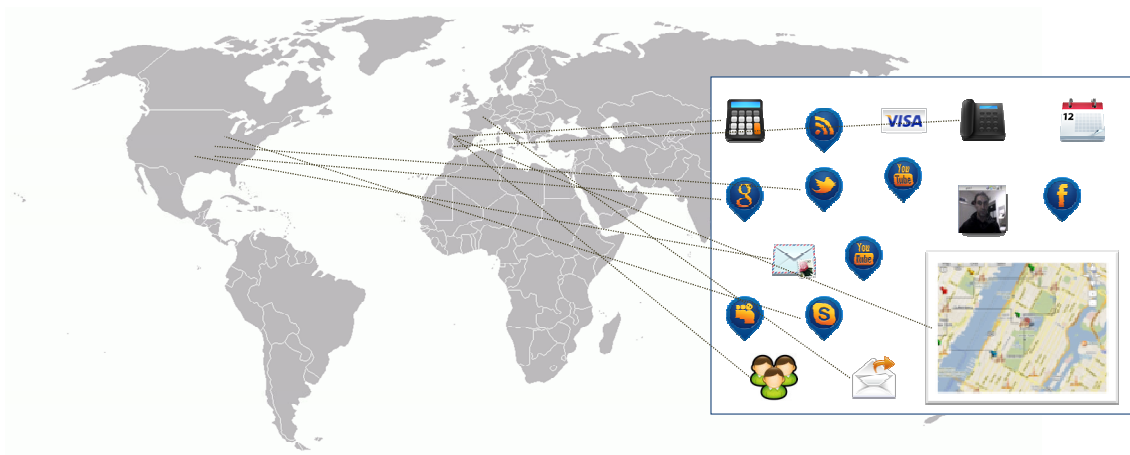


Figure 5.4 - Traditional architecture – several applications need to be managed

In most traditional solutions, or those that result from conventional development methods (Web, Desktop, Mobile), results in a one more that falls on a set of existing solutions and of which you will need, as a whole, to get the implemented services you want. In this context the complexity come from the portability and interoperability between existing solutions, platforms and equipment involved. The usual "it needs to be installed to be able to use" (Figure 5.4).



Figure 5.5 - Communicational Architecture – Transparency and Ubiquity of services

In a communicational architecture that supports pragmatics and ensures ubiquity on services (direct human interaction) it must offer a set of communicational channels seamlessly and transparently. If the service is available it is offered by the solution (Figure 5.5).

5.2 From Transactional to Communicational architecture

In the traditional Transactional Computing – common web applications - the server of the application models databases and allows the users interaction using web-based interfaces. From Syntactic to Semantic and Pragmatic web, the essential changes is around more meta-information to the existent information, towards the enhancement of its usefulness (Spyns & Meersman, 2007). On the other hand, in a Transactional Architecture, the execution of their system follows semantic relations between their processes. The output of a process is semantically and syntactically interpreted by next one, analyzing inputs and interpreting results. A relation based on technology dependence, having methods signatures, syntactic rules and formal invariants, controlled by specific algorithms and typed oriented matching.

The Figure 5.6 represents a traditional Transactional Multilayer Architecture, where layers, patterns and standards API prepare the system to sufficiently support any specified requirement and easily integrate future ones, granting its flexibility, robustness and interoperability. Here the interoperability means interaction essentially, offering enhanced rich interfaces to multiples

devices (multimodal systems), and the human behaves as mere user, outside the architecture. So, transforming this architecture to support effective human interaction is more than to implement new technical enhanced features. Thinking in that way is a reduced technical perspective.

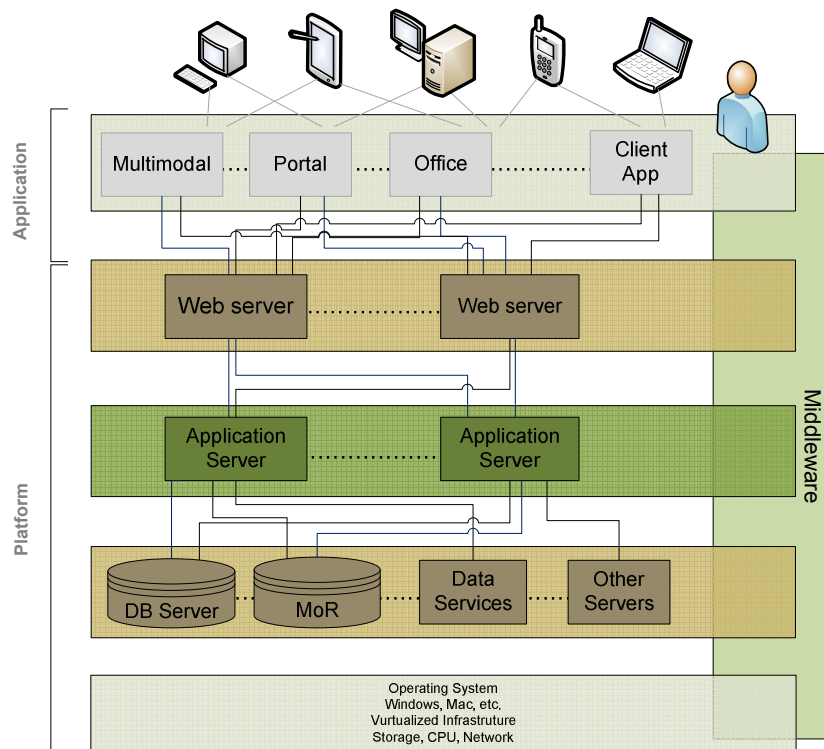


Figure 5.6 - Transactional Multilayer Architecture

Considering this, the real (human) user requirements are not well supported indeed, because they are not easily tangible and technically specifiable. So, the user will need to continue to adapt to the system and follow the system wizards. He cannot have his own reasoning and interact humanly with the system.

All this “formal” behavior results from initial analysis and specification process, requirements oriented, mainly. Moreover, to be relevant to users and business, the system needs to be designed considering both: from the specified requirements and human “preferences” (Microsoft, 2009).

To align the system to human, its architecture needs to support human-to-human real and synchronous collaboration that allows the co-exploration (co-creation) of the system with other

agents (humans). Future architectures need to be communicational based having direct human participation and collaboration in any interested phase. Assuming this, Pragmatic and Collaboration engines allied to effective brokering mechanisms need to be implemented. The evidence of this comes from social networks success and their use for our own interest in a completely autonomous way. In short, a Communicational Architecture represents a Transactional Architecture with Pragmatics services.

Paraphrasing Boinodiris (2010), *“because new media can be used to engage and immerse users in a highly tailored environment, a highly customized perspective about the specific needs of a customer can be developed. Organizations are consequently equipped with much greater insight into customer needs and desires.”*

Larger is the communicational capability of the architecture, greater is the effectiveness of the system.

5.3 Semiotic based Architecture

In the context of a semiotic based architecture two main characteristics are required: a) ubiquity and b) communicational (G. D. Putnik & Putnik, 2010b). The cloud grants the ubiquity of registered resources (services) and innovated and efficiently integrated communication tools will complement the semiotic features.

Even any device can be used to explore existing applications they can behave as a mere tool to interact with the system and not as pragmatic supporting mechanism (Figure 5.7).

The intended communicational architecture (Figure 5.9) we need, in short, to support Transactional paradigm and Pragmatics, means: a) rich client interfaces with sufficient interaction to allow user agility and competence, b) multimodal, for client device classes (Figure 5.10) support and c) communicational to allow pragmatics, where human-to-human real interaction is completely supported, as happens with communication (chat, video, conference) rooms, for instance.

Rich Client Interfaces must grant user accessibility and useful and friendly interaction features, appropriated to the user devices technical capacity (thin clients). They need to support, beyond basic communicational services (text, files, chat, etc.), multiple-user real-time video and audio with a set of auxiliary session tools (recording, whiteboard, away messages, etc.) as well. Most of

these communicational requirements will be supported by implementing appropriate features using existent P2P services engine API SDK, as happen with Skype SDK, for instance.

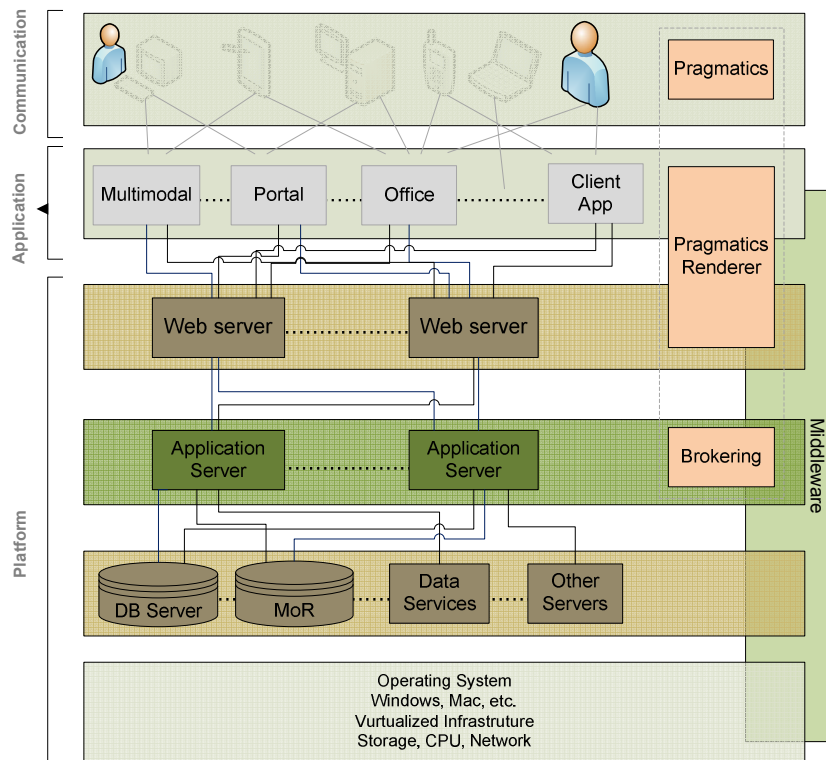


Figure 5.7 - Communicational Architecture with devices “abstracted” as semiotics instruments

The thin client will be implemented with most advanced and powerful open source client device independent technologies, namely HTML5, CSS3, JQuery, XAML and XDIME. To grant asynchronous interaction (by default it is synchronous) and consequent user expression, performance and presence, will be used AJAX Frameworks.

To support multimodal Client Device Classes the thin client interfaces will follow the rich client Web UI programming model (Noyes, 2001) (common *RIA - Rich Internet Applications* model). This model reuses components and skills, while also supports online and offline operations (disconnect scenarios). Since devices have distinct capabilities, physical hardware characteristics and limitations, the client interfaces must be prepared to easily and transparently adapt to.

To reuse components and develop a RIA application, RIA Services following *Model View ViewModel* (MVVM) pattern (Smith, 2009) will be followed. MVVM is a particular case of *MVP - Model View Presentation* where *Presenter* (substituted by *ViewModel*) handled interactions programmatically while in MVVM, those interactions will be handled automatically by the data

bindings (Figure 5.8). These components can be cloud services or already SaaS and must be integrated using an appropriate cloud engine API.

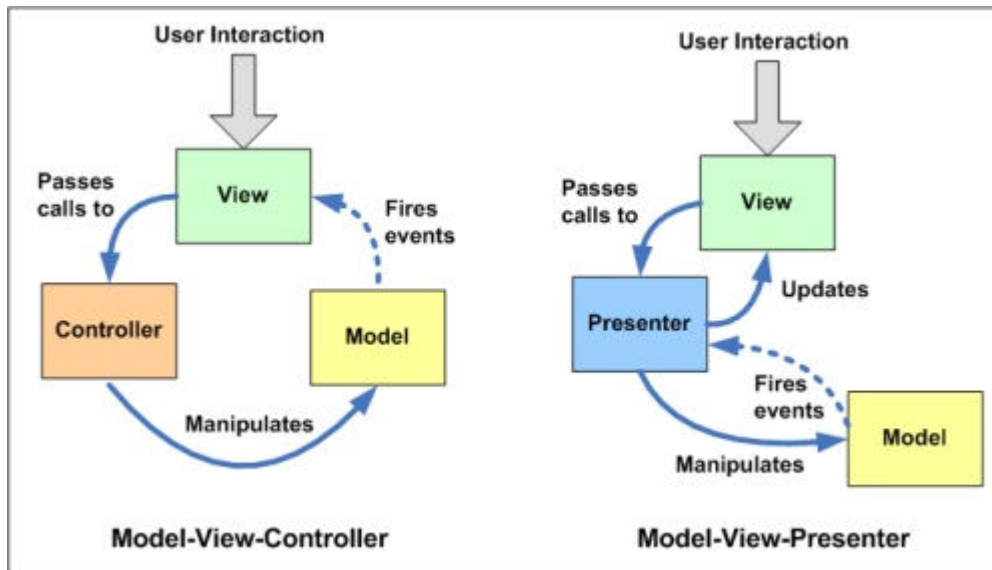


Figure 5.8 - MVC and MVP/MVM models

(<http://joel.inpointform.net/software-development/mvvm-vs-mvp-vs-mvc-the-differences-explained/>)

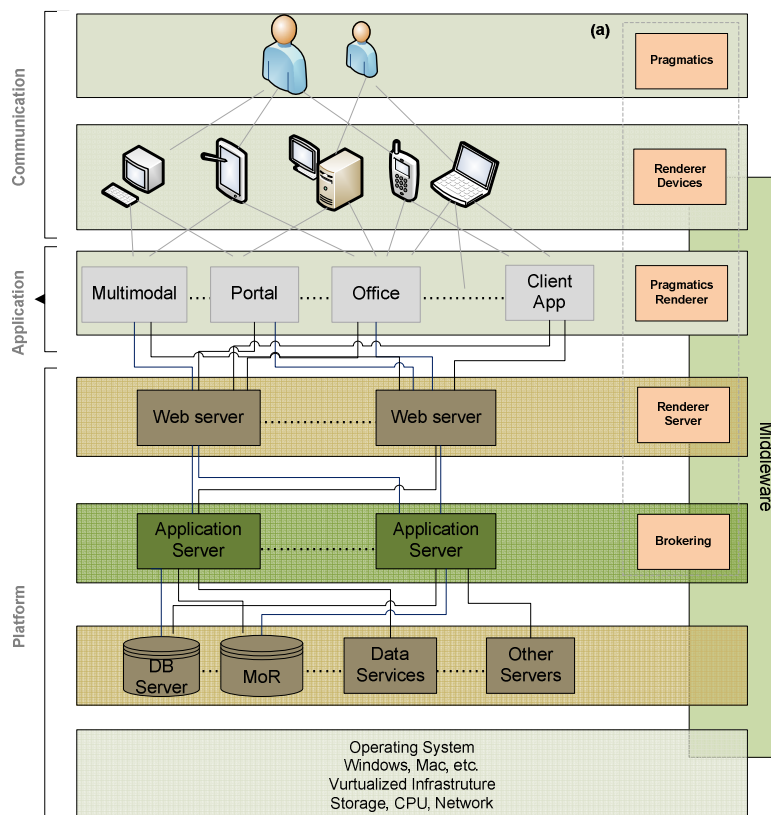


Figure 5.9 - Communicational Architecture where devices are Pragmatics Renderers

Considering Pragmatics, the Semiotic Component (Figure 5.9 (a)), the innovative part of our communicational architecture, it will be organized in three levels: a) device level, which allow user “to use” pragmatics with the system, b) application level which result for a set of tools which allow user a pragmatic based interaction, and c) application server level which is responsible to implement pragmatic engine, the entity which will support all pragmatics services.

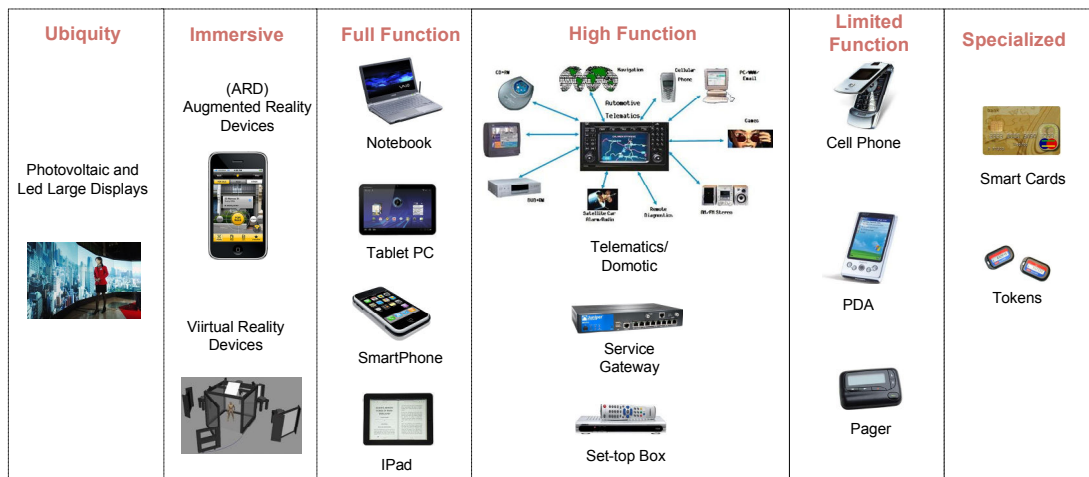


Figure 5.10 - Multimodal interfaces for multiple Client devices classes

(Adapted from (Erl, 2009))

5.3.1 Pragmatic “renderer” component

The Pragmatic renderer works as a communication enabler. It will consist of a set of integrated technology which makes the bridge between the user/devices and the “system”. As described above, pragmatics allows human direct interaction, following his needs and interpretations.

Assuming this, our architecture will be provided with collaboration mechanisms, under synchronous bidirectional channels, and multi-user sessions with recording and historical support. Real-time video, chat, direct visual talking, rooms, spaces, etc., will be some of the enabled services.

Having cloud services as the main supporting architecture, the use of cloud engine API will be determinant to develop a federated or community cloud. The cloud services will be developed using MVC pattern (using WCF or J2EE) and should provide a RESTfull API to support their use and compositions. In this context, pragmatic supporting services will behave as SaaS in the cloud.

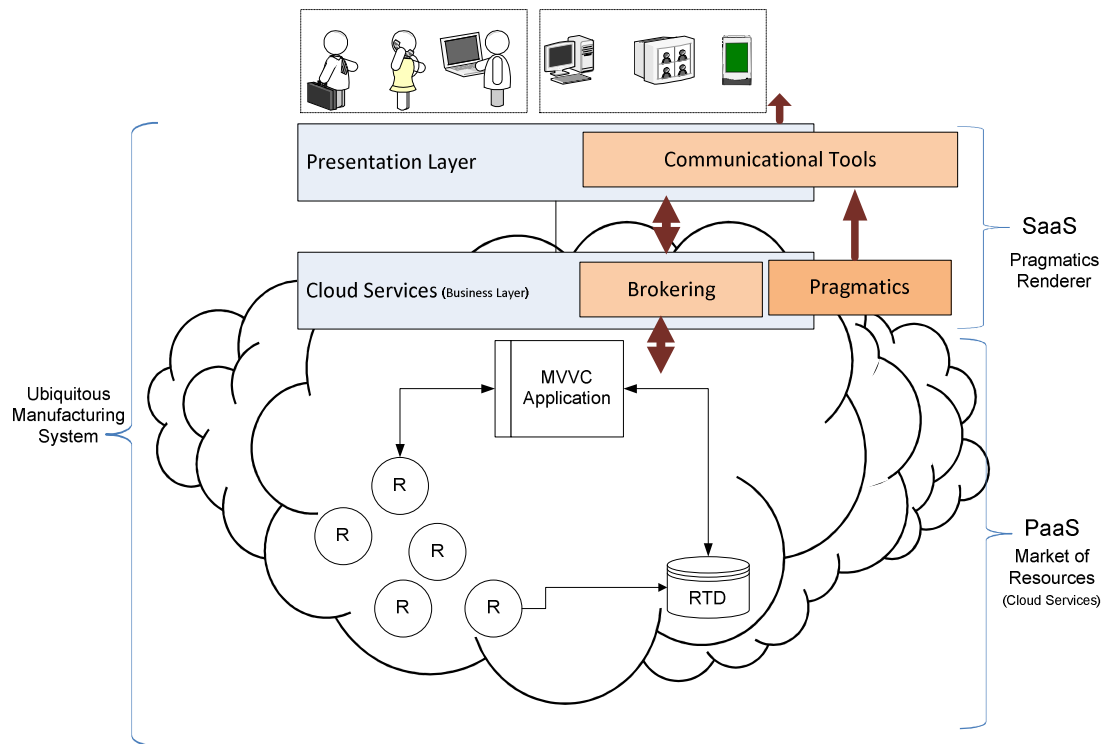


Figure 5.11 - Effective Cloud based Semiotic Architecture

The communications services will be supported by existing P2P technology mainly and the direct interoperability will be granted by existing solutions as Skype API, Google Talk API or others. Innovative communications services will be explored using *OpenCV*, *XMPP* or *SignalR* technology.

The registration and services discovery will be in charge of the API cloud engine. It will support an advanced brokering mechanism over registered services which represent the Market of Services (Resources). Dynamic Reconfiguration and Ranking are two of the multiples features that the broker needs to support. This broker will be implemented using cloud computing model and code behind should be used following Web Services programming model. In practice, this Market of Resources will behave as a PaaS (Figure 5.11)

5.3.2 Cloud Architecture towards Ubiquity Manufacturing

As happened with multiple economic activities, traditional manufacturing has been hardly “shaken” to efficiently integrate ICT in their processes. Nevertheless, efforts to modernizing legacy applications (or initial investments) and to capitalize traditional knowledge still continue to slow down an efficient ICT adoption and consequent business model changes, essential

requirement to re-align with new market requirements. In all this process the human has been a passive actor and the knowledge does not represent the real human capital. Indeed, following working processes (flow) and responding to system's events does not allow the co-creation of knowledge.

5.3.2.1 More than an innovative business model

ICT has brought new economic and commercial relations and provided a global (virtual) market, with new entities and new rules. Agility and quickness are critical in nowadays competitiveness requirements. "*Globalization, innovation and ICT are transforming many sectors to anywhere, anytime platforms*", towards an intelligent business model under "*design anywhere, make anywhere, sell anywhere*" paradigm (Elliott, 2010). We would add "anytime" too. Traditional suppliers and customers are "transformed" in services, where supplying or using profiles are a question of needs or context. One service (a *Calculator*, for instance) can execute (supply) something using other services (*Add, Sub, Mult* and *Div* operations) (Usmani, Azeem, & Samreen, 2011).

Manufacturing has been looking for low cost processes and scalable resources capacity. These resources, even in a global market, must be discovered, selected and managed, and the capacity and efficiency to get the "best" ones will be determinant. Some intelligence must be put in this process (Mostafaeipour & Roy, 2011), but it is not enough to achieve the expected efficiency and sustainability. Ubiquitous Manufacturing sustains the needed agility and quickness to react to market changes (G. D. Putnik, 2010; G. D. Putnik & Putnik, 2010a).

Although being autonomous resources (services), i.e., projected and created "to work alone", these resources could not be sufficiently integrated or integratable (Mackie, 2007; Singh, 2003).

However, not only these technological trends influence the course of the things, social-economic trends as consumption growing, globalization, innovation and sustainability policies, determine new orientations too (Majumdar & Szigeti, 2011). So the challenge is not only the ICT adoption but more the way one does it. "*(...) the biggest problem is not the availability or implementation of technology: it's changing the mindset of the people themselves.*" (Elliott, 2010).

5.3.2.2 ICT, Dashboards and Cloud Manufacturing

Cloud computing is much more than unlimited IT capacity (processing, hosting, etc.). It is an opportunity to achieve, indeed, new business models.

The Web will continue to be the main channel to support business activities. Furthermore, the success of human capital promotion with social media, and the new communicational (smart) devices capacities, brings web (wired or not) to high levels of intelligence support, leaving far behind the initial syntactic hypertext model and its semantic content models successor. Web (3.0 and 4.0) (Figure 5.12) will support value creation and (self) efficient business models to use it (Bhakdi, 2010).

Basically, Cloud computing success arrives from its capacity to support providers requirements, services-oriented infra-structures and economy requirements, emergent (virtual) enterprise requirements, and user requirements (Xu, 2012). However, interoperability, security and QoS details involving all “stakeholders”, including distinct cloud models, bring this “success” hard to get and questionable (Mulholland et al., 2008).

Cloud Manufacturing represents a shift from production-oriented to service-oriented manufacturing, being services IT instances of (traditional) resources. Thus, the existence of efficient protocols and APIs (Application Program Interfaces) to manage cloud services (Services Oriented Architecture (SOA) Governance area), easily supports the required dynamic resources allocations and coordination of cloud Manufacturing (Bo-hu et al., 2010).

Cloud-based application architectures (Betts et al., 2010) present a “transparent” layer between presentation layer (client interaction) and business and data layers (business rules and contents in cloud, mainly).

Application’s Presentation Layers are now structured in a set of widgets (cloud-based full-fledged applications (W3C, 2011) or *cloudlets* as e.g. *Podio*⁴⁰ specialized work apps or *Google Apps Marketplace*), each one, owning its graphical representation, support a service that can easily be “composed” (integrated) in a dashboard “*expected to improve decision making by amplifying cognition and capitalizing on human perceptual capabilities*” (Yigitbasioglu & Velcu, 2010). Despite of this, the components are not effectively integrated, but merely functionally organized, indeed. They do not interact. Furthermore, besides the restricted interoperability, the lack of

⁴⁰ *Podio* is an emergent dashboard like online work platform

effective and really integrated communicational instruments, essential to enable the user participation (embedding his experience) on decision processes, represents another important weakness.

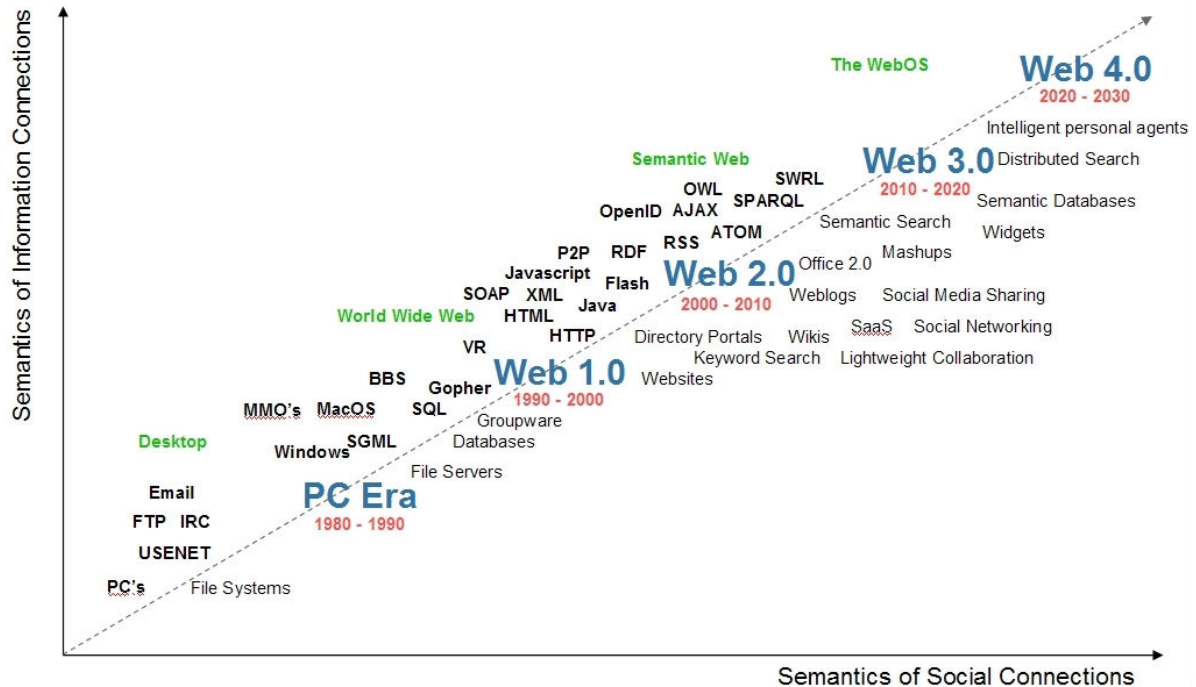


Figure 5.12 - Web 4.0 brings intelligence⁴¹

ICT will continue to behave as a lever but the way technology is used might be more important than the technology itself. Technology generates legacy.

5.4 Communicational Architecture

The communicational architecture here proposed guarantees a platform for interoperability between services and multiple channels of communication among participants. Is based on the following assumptions:

- It Is oriented to networked computing, usual cloud computing;
- Offers technology independence
- Ensures scalability

⁴¹ Radar Networks & Nova Spivack, 2007

- Ensures easy integration of new components
- The communicational services arrive from services suppliers
- Behaves as services middleware.

5.4.1 Conceptual Model

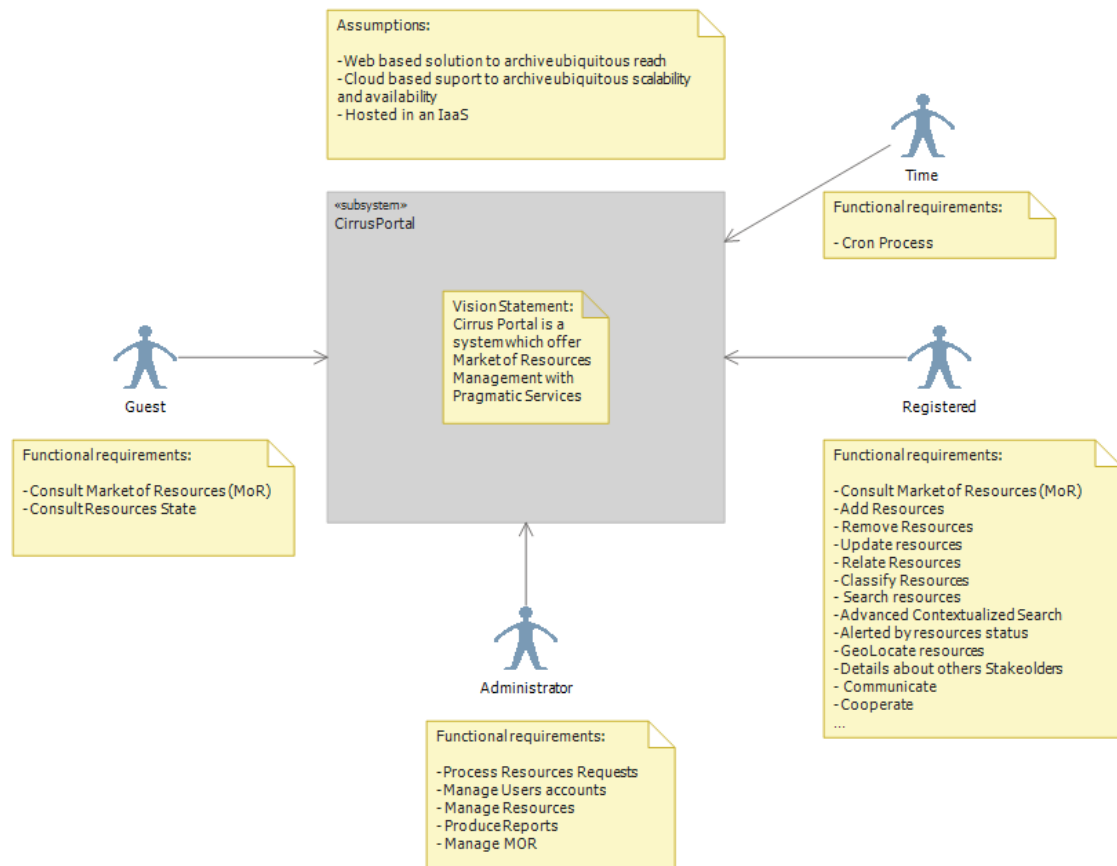


Figure 5.13 - Context Diagram – Scope and Profiles

The architectural behavior is managed by typical rules of web 3.0 applications or social networks, i.e., their sustainability are the responsibility of users themselves. As with most common social networks (Facebook, Twitter, Google+, etc.), the infrastructure has mechanisms that allow membership (the registration) to the network autonomously and independently. A new service appears if any entity has registered it. The state of the service (operational, busy, disconnected, etc.) manifests itself according to the intervention of its promoter.

A service can be seen as a resource that can be used by others. It can be a person, a machine, a set of tasks (an activity), a simple task, etc.

The architecture involves the registration of new services. Its management is partly the responsibility of the promoter, and the platform just the responsibility of its integration into search engines (*brokering*) and rating (*rating*), basically.

In a broader perspective, consists of a network of resources, properly graded, with selection and use rules (*Market of Resources*), and a set of monitoring tools that allow to monitor the status of different resources and their participation in activities to which they were associated.

Objectively the architecture involves the participation of three entities (*users*): a) *Registered entities*, which essentially comprise the *Service Provider* and *Service Customer*, b) invited Entities (*Guest*), who only can browse the system; and c) Administrator, who is in charge of managing the whole system (Figure 5.13).

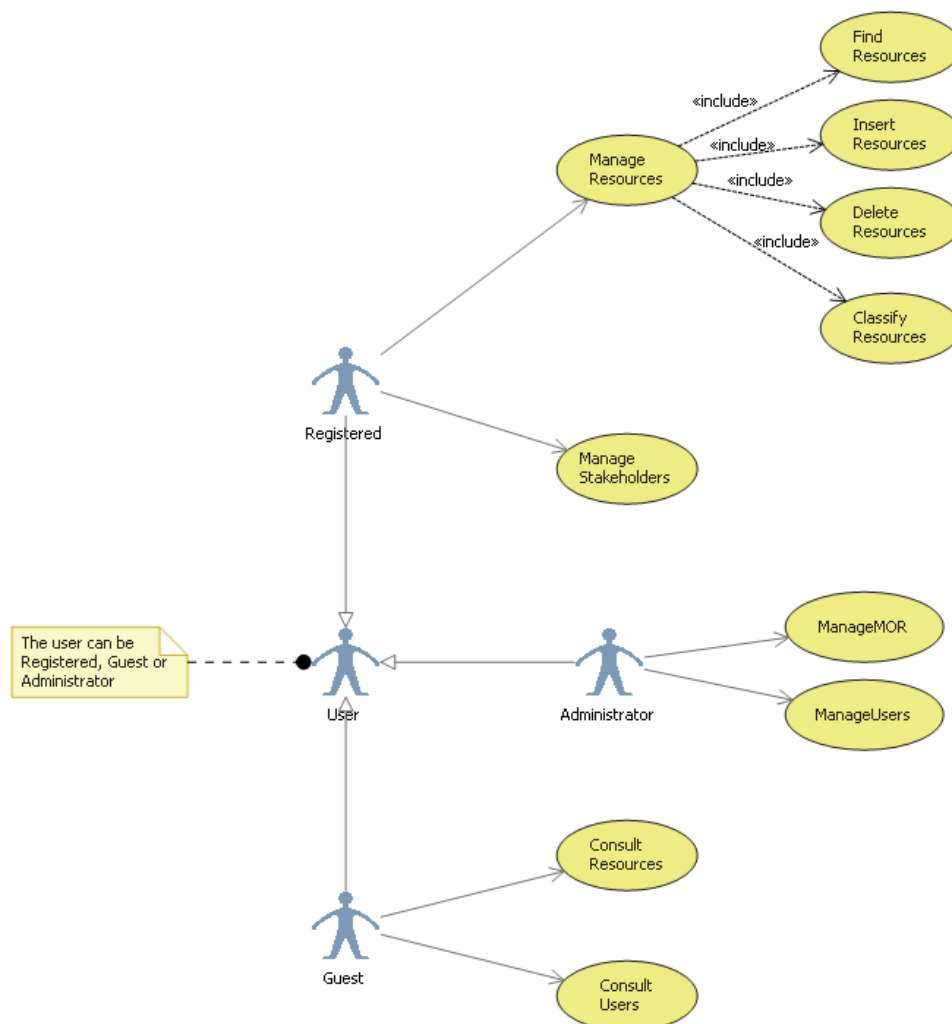


Figure 5.14 - Actors and Uses Cases

Each entity (*user*) has his own business-value, with an area of action (*scope*) and profile (*actor*). The use of the system by each of these entities is thus contextualized to the profile and scope.

Figure 5.14 summarizes the most relevant Use Cases associated with different profiles (*actors*), internal or external to the system, representing the main system functions (features) and the roles or responsibilities that each player has on them.

In view of the Domain Conceptual Model, the diagram in Figure 5.15 shows a subset of involved entities and their relations, as well as some enumerated values.

Each service provider (*user*) promotes a set of services supported by their resources properly registered in the Market of Resources. Furthermore, each resource has a set of indicators that monitor his real state.

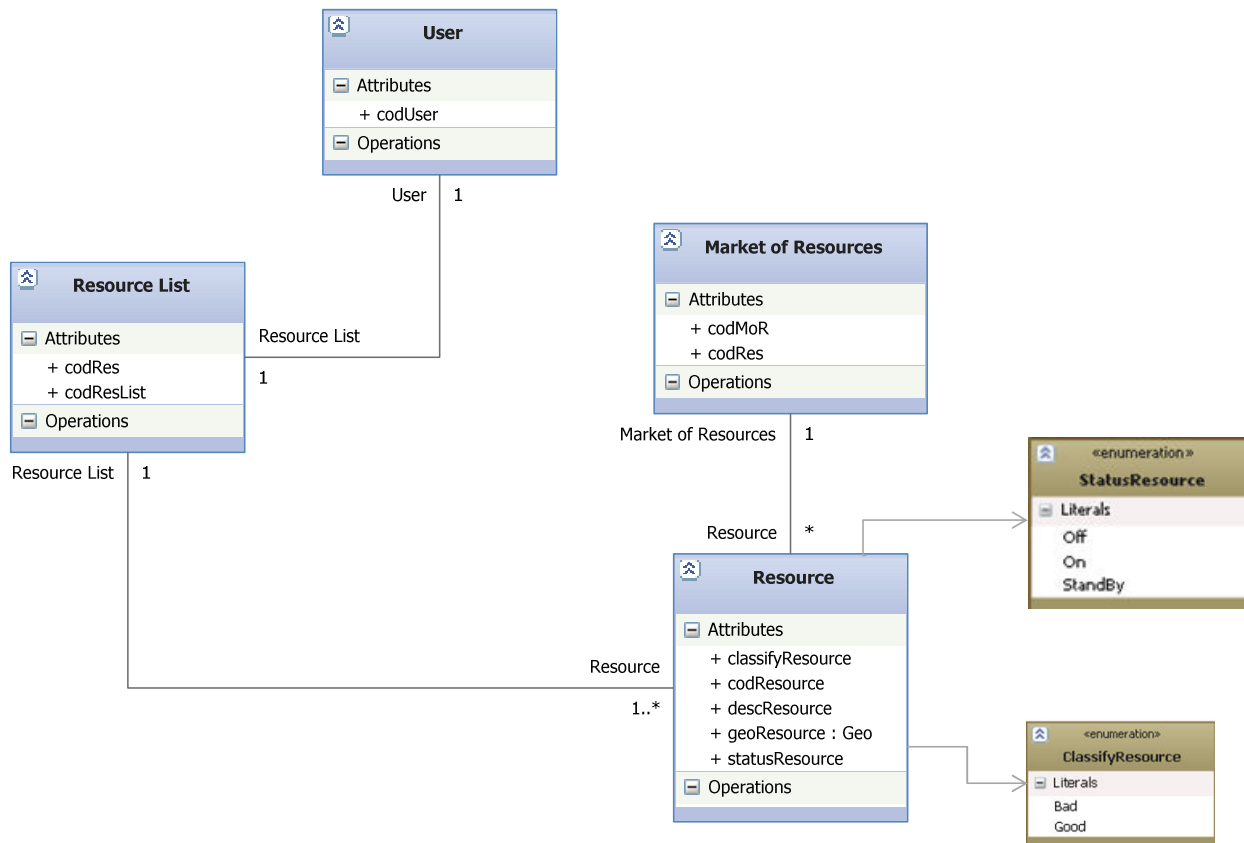


Figure 5.15 - Conceptual Domain Model

(Edmonson, 2010)

It is obvious the importance of the service that is provided and the resources that support it. Following the intention to have a communicational network, each resource has its own communication services (that can provide). So, in addition to several other information

(specifications, history, calendar, information, other), each resource is also expanded with a set of Pragmatics Channels, that ensure pragmatics in the system, i.e., the "natural" interaction between participants (Figure 5.16).

5.4.2 Logical and Functional Model

A logical model is always associated with a functional model which is expected to be implemented. Functionally it is intended to develop a platform with social engine type behavior, similar to facebook, google or other engine, able to:

- Have autonomy in relation to the technological requirements of the participants;
- Have autonomy in relation to decisions of the participants;
- Each participant manage its "context".

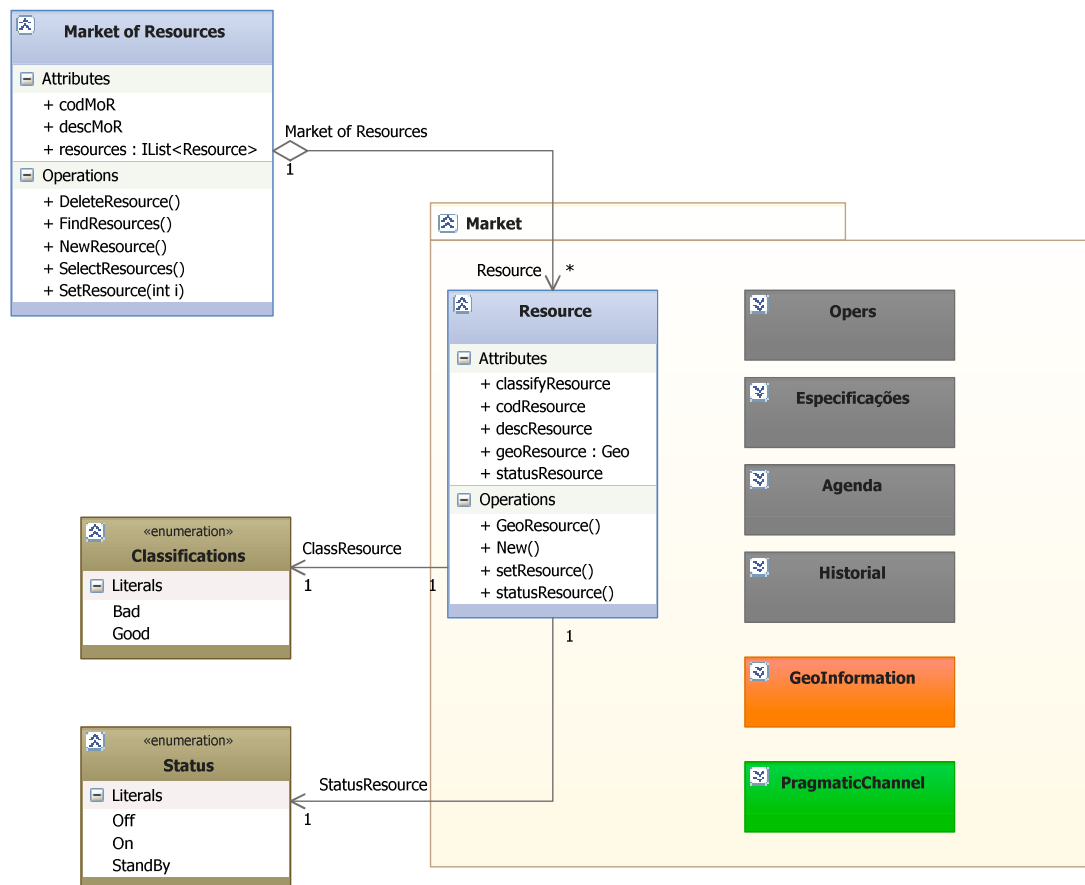


Figure 5.16 - Pragmatic Channels

- The overall management of the network of participants is sustainable because results from individual management context.
- Offers a set of services (tools) that the participant may use

These features are resulting from the operation of various components, whose responsibilities are logically distributed. Thus, the architecture results in a logical set of several components, of which we highlight the three main:

- Repository
- Broker
- Pragmatic Renderer.

Each component "responds" appropriately to the different users of the system, *Service Provider* and *Customer*, users who ensure the registration and use of each resource. The service provider may still prefer to have a proxy, represented here by *Agency*. This *agency* can be someone which represent a specific manufacturing machine or can match a tourism agency that intermediates between the tourist and the promoter (Figure 5.17), for instance.

On the other hand, each component of the architecture presents its properties and responsibilities. The Resource entity reveals itself as the cornerstone of the whole architecture, it is the base element (*data type*) and its definition is accomplished via attributes/values tuples, complemented with other semantic data. All figures follow existing domain ontology. For example, the power of a machine has an attribute "watts"; or a travel has an attribute "distance" to which is associated a numerical value.

In an attempt to use a certain resource, the desired information relies on the existing register of that resource or may result from the co-creation accomplished through the communication channels which *Pragmatic Renderer* manages. In practice, in the case of complementary information needed for a particular resource (if it is operational, busy, etc.), it can be achieved by interacting directly with his provider.

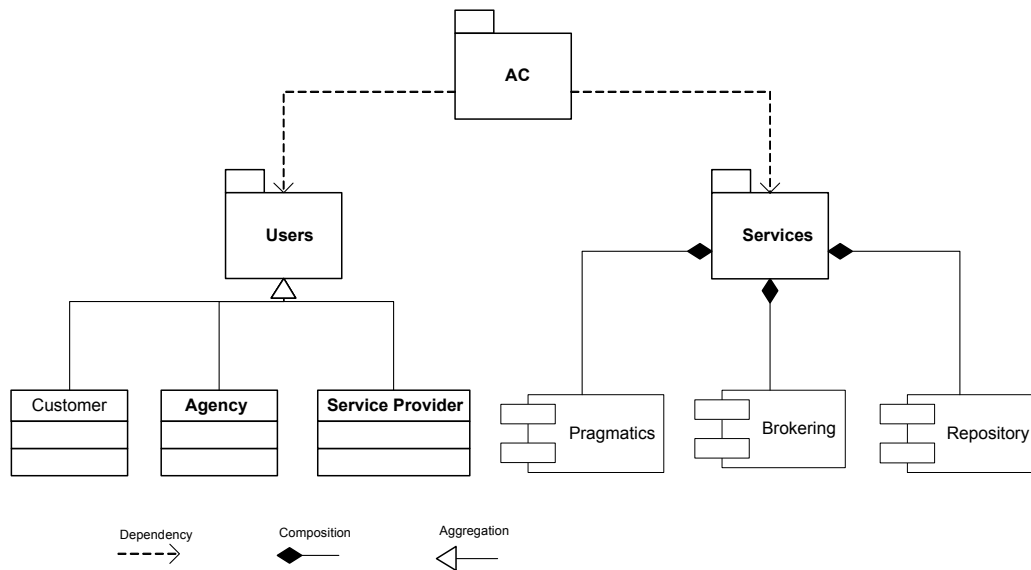


Figure 5.17 - Components and Entities Logical Model

The identification of the desired resource is only possible through appropriate selection procedures (*Brokering*) in the resource repository. This selection is conditioned by multiple factors, since the availability and quality of performed services, up to the user's requirements. To bridge the gap between the interest of the user and the information that it wishes to, the search engine is still sensitive to context and to the own profile of who is searching. The decision on which the resource to choose is moderated by the end user.

As a whole, the set of resources determined by the execution of a particular task constitutes a Virtual Enterprise, whose sections, although autonomous, directly or not, collaborate with each other.

For all this to be possible, each resource has been previously and properly registered in the *Repository* (the Market of Resources base), being the registration process autonomous and completely independent (in time and context) of the selection process. The registration follows automatic or explicit mechanisms (manuals) of cataloging or classification, using as meta-information terms of a domain ontology. For the classification of the resource, the user feedback will be important, too.

Thus, the *Broker* works: a) as a tool to help the selection and composition of resources that best satisfy the requirements and expectations of the user and b) as a tool to help on the dynamic reconfiguration management, inherent to the constant change of requirements and the state of resources, ensuring the better alignment between the virtual enterprise (VE) thus formed and the task(s) to execute.

The dynamic reconfiguration of resources is due mainly to: 1) performance and availability of *providers*, 2) changes in user requirements (and therefore changing the task as a whole) and 3) not controllable external factors.

The *Pragmatics Renderer* component is also relevant when the participation of other registered users (or not). They may obtain information directly from the resource promoters or even be followed in decision making. For example, the process to find a resource that executes a particular operation that is far for a particular distance from the point where you want to use.

The quality of service is necessarily the most important factor in resource selection. In practice the user wants to see well executed given task and do not want to worry about how it is performed. As an example, on a vacation trip the tourist wants to go out, have fun and come back, with all the guarantees of good quality in the performed service. To register this quality, additional semantics information must be appended to each resource.

Faced with unforeseen situations (external factors), the system should reconfigure itself to ensure the total satisfaction of user requirements.

5.4.3 Technological Model

In a global technology perspective, the architecture must support secure real-time collaboration and synchronous or asynchronous integration between processes.

5.4.3.1 Hybrid Cloud based

At a time when there is the challenge to develop *Software as a Service (SaaS)*, the distributed computing of companies (increasingly more into the Web) tends to restructure to a model where cloud-based infrastructures prevails, leading to their service offerings are (re)implemented (*dashboards, cloudlets, etc.*) to this new context.

The same happens with this architecture. Although the adhesion to the cloud suffers from the same syndrome that famous adhesion to the eCommerce suffered, i.e., although conscious of the economic benefits drawn from it, there is some resistance to the loss of some control over what "we already have", it is certain that the evidence already given by the cloud computing infrastructure in which issues such as usability, scalability, flexibility, and others, show that the bet is meritorious.

On the other hand, if you want a scalable solution, where the key word is ubiquity of services, and whose sustainability is derived from user participation, i.e., cannot be fully controlled by us, the cloud is challenging, interesting and relevant.

Aware that an economy of scale entails some loss of freedom, and that the optimization is achieved with the expertise, we see the cloud as a specialized system, with few degrees of freedom compared with the dedicated development, but which offers a high economy of scale.

In the future, the context in which the architecture is applied (Tourism, for instance) foresees an exponential membership of users, being then critical the responsiveness of the infrastructure. Taking the context for the Manufacturing area, the amount of resources (equipment, operators, services, etc.) will be even higher.

Since we want to maintain some flexibility and agility, mainly in stakeholders's side, we are conscious that not all services will be cloud-based. So a bet on a hybrid strategy guarantees, on the one hand, an economy of scale and ubiquity, and some control considered essential, on the other. The information systems of the multiple stakeholders are not preventing accession and the decision to "be or get out" is for them. We focus a little on buzzword *Lowering transloading cost in the context of software architecture: localized optimization through selective specialization (LOtSS)* in which the company optimizes its services deciding what to develop internally or adopting existing solutions.

5.4.3.2 Structure

It is a structured layered architecture derived from the standard *Event Driven SOA*, a hybrid participation of *Event Driven Architecture (EDA)* standards with *Services Oriented Architecture (SOA)* (Maréchaux, 2006), duly integrated into the cloud.

EDA architecture enables the transmission of events between components or stand-alone services, allowing, for example, the asynchronous pushing in the repository of resources from providers, or even the reaction to reconfiguration triggers. On the other hand, the SOA architecture supports the discovery (*brokering*) and composition of resources, ensuring efficiency, portability and agility of technology.

If you need integration with the providers's information systems, the adoption of the standard *Enterprise-Service-Bus* (Maréchaux, 2006) will support this process.

The brokering of resources on the Market of Resources is guaranteed by the use of WCF services integrated into the cloud (*WCF Cloud Services*). Together these services provide an *REST/SOAP Application Programming Interface (API)*, which supports synchronous or asynchronous behaviors, transactional (or not), and ready to be integrated in any application.

In addition to brokering, the API still has services for resource management (registration, removal, change). In this way any external application (on any platform or technology) can be easily developed and thus integrated into the system (Figure 5.18 a)).

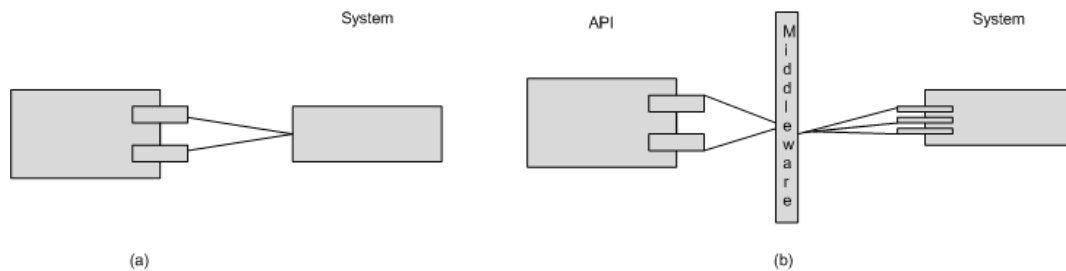


Figure 5.18 - External Systems Integration

In the situation where a user (customer) need to interact with any other(s) (customers or providers), there are integrated communication services (video, audio, etc.) in real-time, whiteboards for collaboration, etc., that each resource makes available, ensuring the proper communication between people.

The layered structure resulted from the combination of the *Model-View-Controller* pattern (MVC) (Figure 5.8, pag. 101) (Hasan, 2010) and *Rich Internet Application* (Microsoft, 2009; Preciado, Linaje, Comai, & Sanchez-Figueroa, 2007) (Figure 5.19).

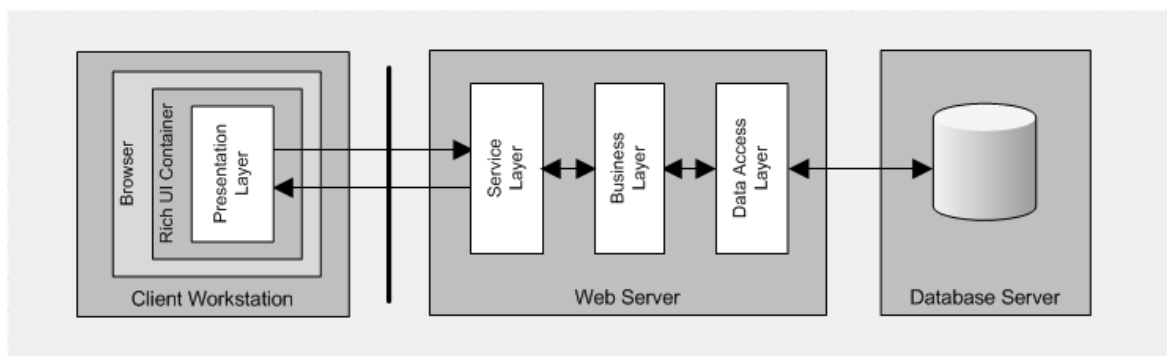


Figure 5.19 - RIA Pattern

(Microsoft, 2009)

The reference to WCF (cloud) services is distributed in practically all the logical components of the system, both at the presentation layer (*View*), business rules (*Controller*) or even in data access (*Model*) (Figure 5.20).

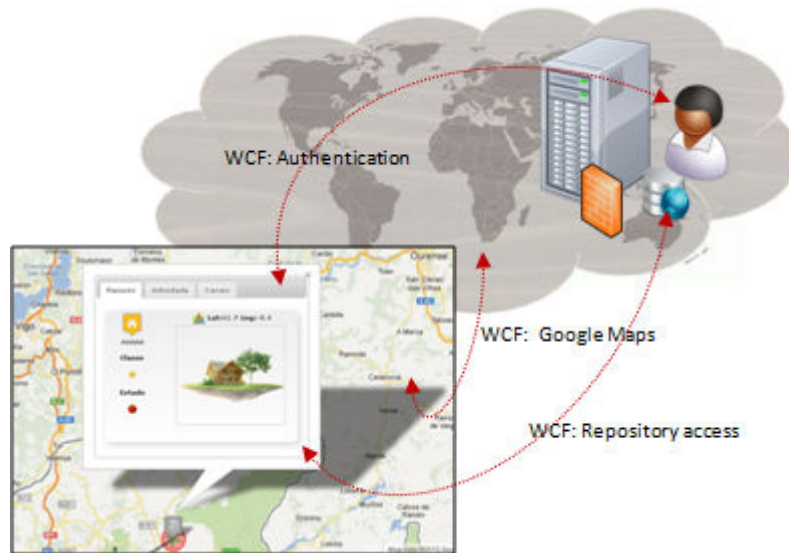


Figure 5.20 - WCF Cloud Services

For example, we find *cloud services* at the presentation layer with Mashups that use cartographic services of Google Maps.

Since the database is hosted in the cloud and has a WCF API with all services for its management, its use by any other application is done through cloud-services in its data layer.

Even the authentication services, inserted in the business rules layer, use cloud-services since the registers in the system network was made in the cloud hosted database.

The Figure 5.21 represents the global architecture where they show all the components, structured by the responsibilities they have, either by the interaction they establish among them. It can be seen clearly a part that is supported on the server side (web server and cloud server) and another that is supported on the client side, being it a mobile application or not, to use (*Client*) or to manage (*Manager*) the system.

Whenever it is necessary to integrate the various layers, a transverse layer takes care of global services such as Security, Operational Management, Communication and other.

The integration of external applications (*Services Consumers*) is possible thanks to the developed web services (*Cloud Based Services*).

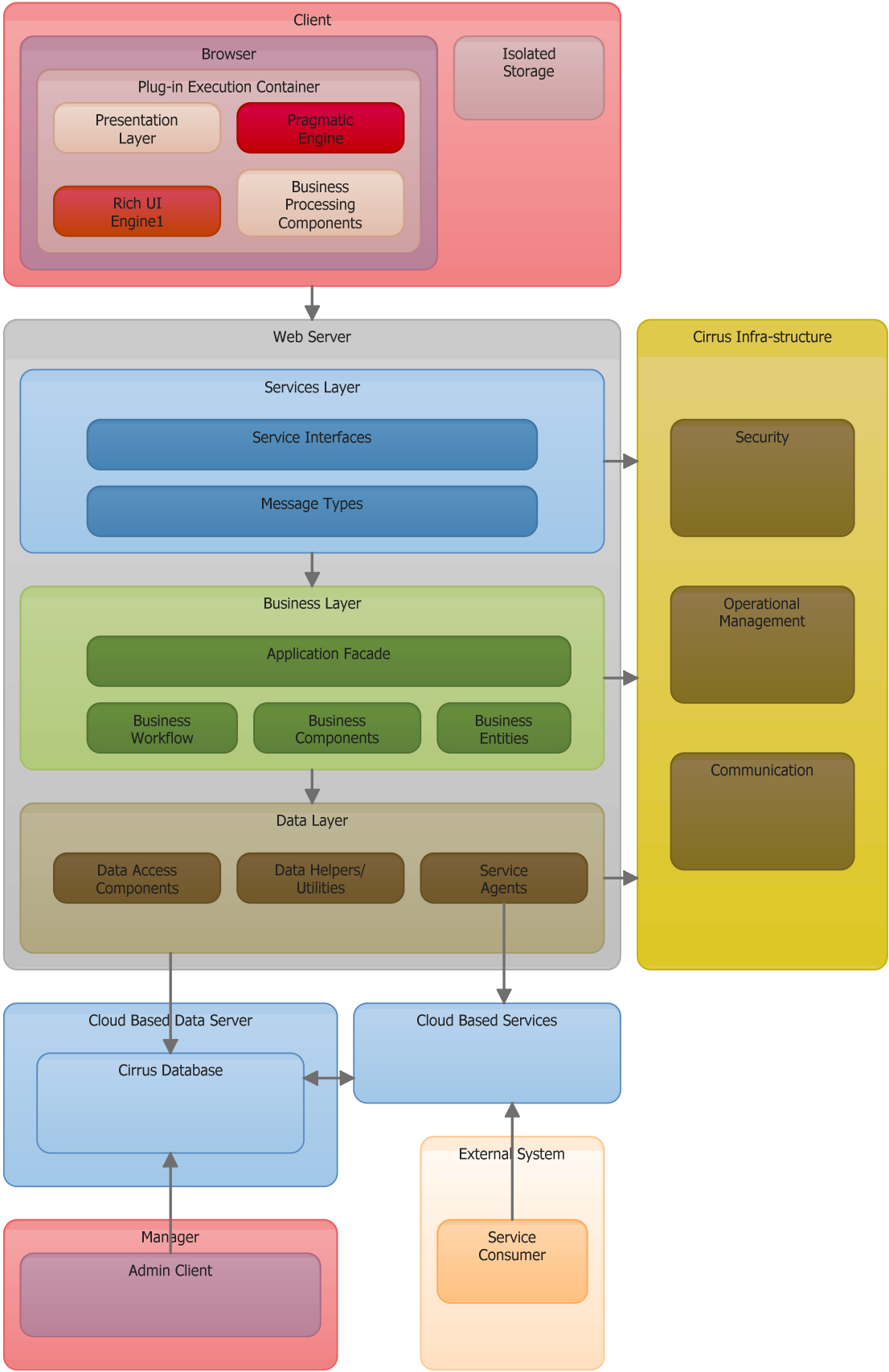


Figure 5.21 - Technological Architecture - MVC/RIA Pattern

System users (*customers*) interact with the application through the *Presentation Layer*, and directly between them through the services provided by *Pragmatics Engine* component. External systems interact through the layers of services, whether they are in the cloud (*Cloud Services*) or in the application itself (*Service Layer*). Both layers, *Presentation* and *Services* must "comply with" the rules implemented in the Business Layer. This rule is maintained by the *MVC Controller* component.

An external system can be another web application, a mobile application, a mashup to integrate in a dashboard, etc.

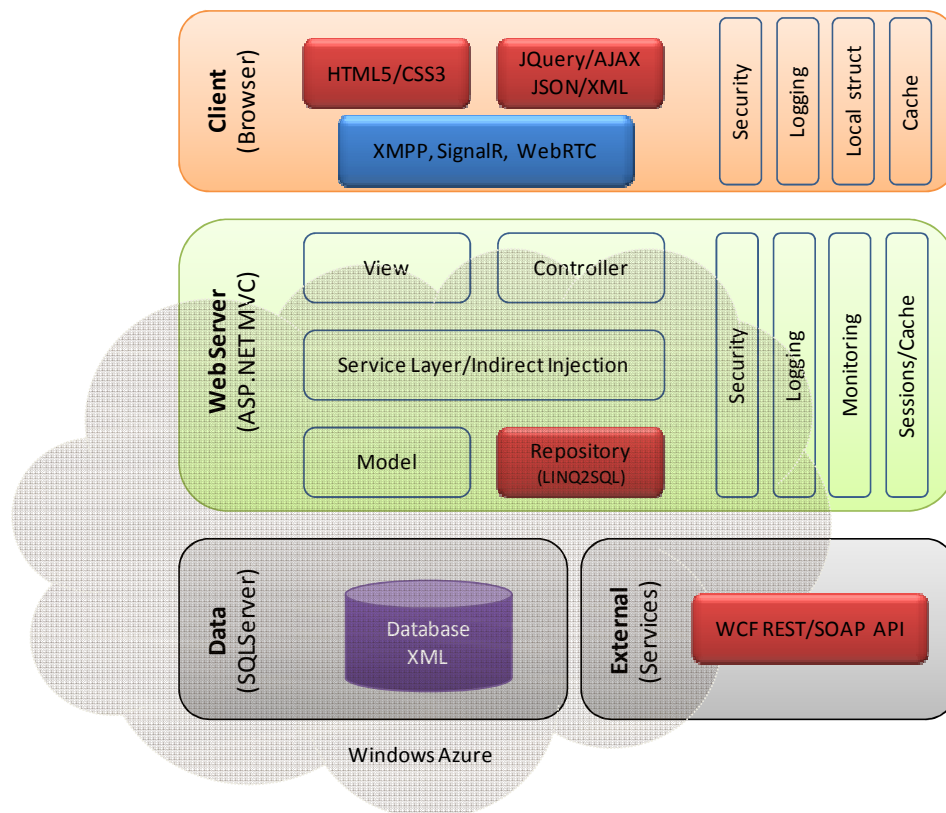


Figure 5.22 - Technological support

Figure 5.22 describes the technological support associated with each of the layers/components, highlighting the technological component that supports the *Pragmatics Engine*. This is a set of services that each resource offers that allow it to establish a direct communication channel, synchronous or not, with collaborative nature or merely one-way. We refer to services such as *email*, *chat*, *video conference*, *chat rooms*, *white boards*, etc. Considering several existing technologies, and focusing on the main ones that offer real time synchronous communication, such as *XMPP*, *SignalR* and *WebRTC* (*HTML5 WebSockets Protocol*), show that has not yet

reached a satisfactory capacity of interaction, especially if we refer to non-proprietary technologies, as has been the case.

In short, the architecture was implemented with the emerging technologies development for Web 3.0, including:

RIA Web Pages (View)	<i>ASP.NET, HTML5/CSS3, JQuery, AJAX, JSON, XML</i>
Asynchronous requests	<i>AJAX</i>
Parallel requests	<i>Threads/SignalR</i>
Server Business Logic (Controller)	<i>.NET (C#)</i>
WCF cloud Services	<i>.NET(C#), XML, JSON</i>
Repository (Model)	<i>.NET (C#), LINQ2SQL</i>
DataBase	<i>MS SQL Server, XML,JSON</i>
Cloud "housing"	<i>Windows Azure</i>

5.4.3.3 Pragmatics

As the aim is the creation of a communicational architecture, the communicative dimension with the system or between users of the system has an important role.

A cloud-based architecture must abstract from technological constraints. It is not possible to require all users to use as e-mail the service *Microsoft Outlook* or *Skype* as chat service.

To keep this "independence" and thus ensure the portability and flexibility required, the platform only serves as an intermediary for some of the services (e-mail, for example), counting for this with the cooperation of the browser or web server, when the necessary redirection to the application that executes this service. However, if the device does not have any application that allows sending e-mail, the platform incorporates this service and enables in their use.

To avoid platform dependencies, i.e., has a particular application installed or not, the services will tend to be supported on internet protocols such as HTTP, WebSockets or others. For example, the video chat service may be supported via *WebRTC*.

The architecture provides mechanisms that centralises and disseminates all communicative channels that each resource has indicated during your registration on the Market of Resources. The status of each channel (on, off, busy, etc.) is the responsibility of the provider of the resource.

In practice it is intended to avoid having to execute many different applications to accomplish multiple communicative channel (Figure 5.23 (a)) and instead, in a transparent and integrated way, when using the Web Portal developed, all channels "open" are viable to use (Figure 5.23 (b)).

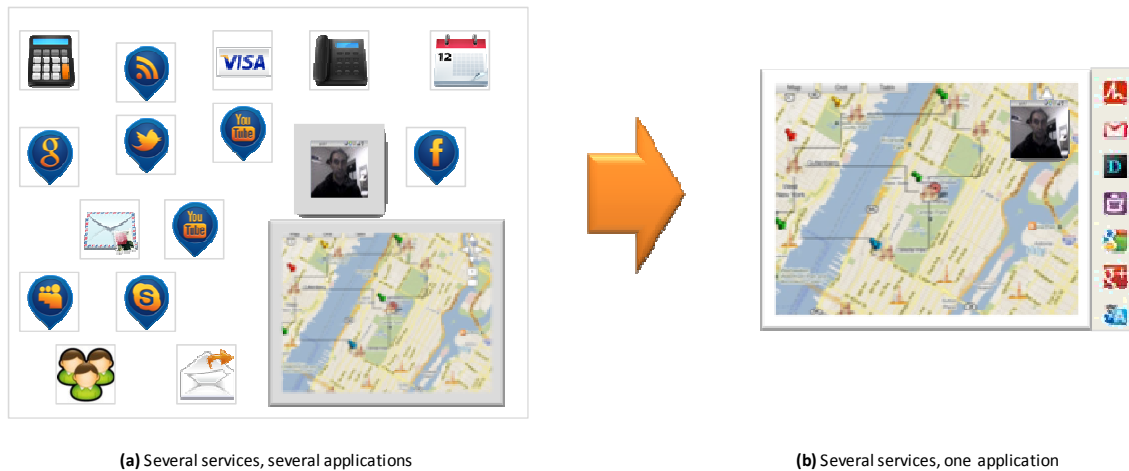


Figure 5.23 – Integrated Communications Channels

5.4.3.4 Summary

The diagram of components of Figure 5.24 shows the physical structure of the system, its main parts, and the interfaces of components that translate on the services they offer or require.

Initially simulation platforms as *OpenSimulator* or *SilverLigth*, as well as *Augmented Reality Interfaces* (OpenCV, Blender, CUDA) (Cawood & Fiala, 2008; Höhl, 2008), were intended to be explored to integrate. Because we intend open source platforms, *Elgg*⁴² and *Exo*⁴³ were social cloud engines to be explored too. Due to time limitations such intentions could not be explored and must be considered in future works.

⁴² <http://elgg.org/>

⁴³ <http://www.exoplatform.com/company/en/home>

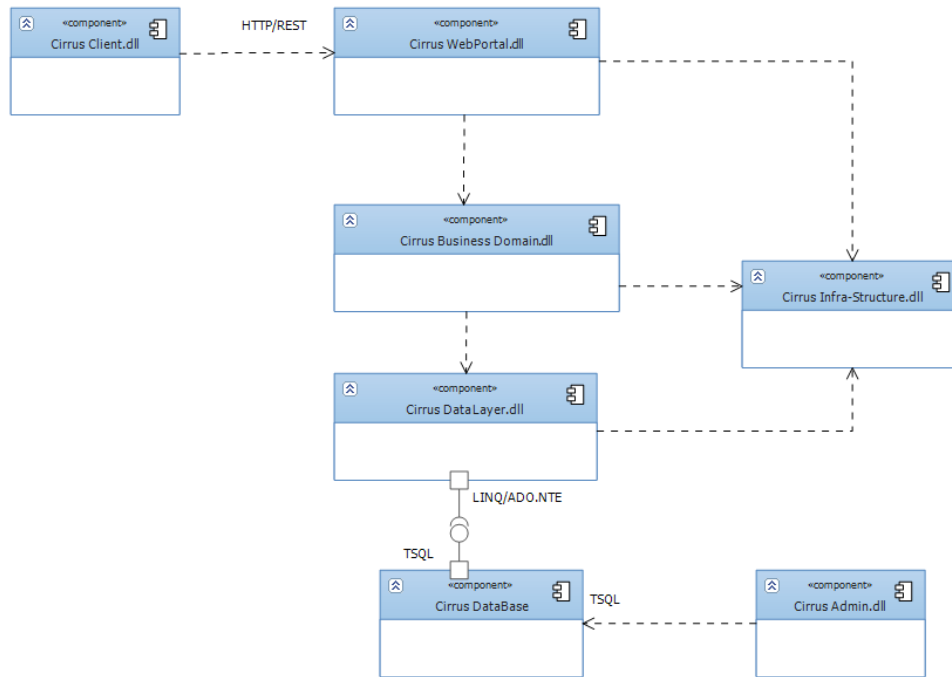


Figure 5.24 - Components Dependencies Diagram

5.5 Cloudlet Architecture Dashboard

Considering the mentioned problems, we propose an integrated architecture which sustains the management and coordination of cloud-based services (resources) to grant technological integration requirements, as well as communicational instruments, as pragmatics tools to support human-to-human interaction, granting effective user participation (Ferreira, Putnik, Cruz-Cunha, Putnik *et al.*, 2012).

This base architecture follows *Model-View-Control* (MVC) pattern and the interface follows the *RIA Presentation Design Pattern* (Cunningham, 2003), having resources and their governance services (Model) hosted in cloud (Figure 5.25 (a)), cloud-based Representational State Transfer (REST) services to support business rules and actions (Controller) and multimodal *Rich Internet Application* (RIA) Presentation Layer (View) to allow multimodal device interaction with (Figure 5.25 (c)).

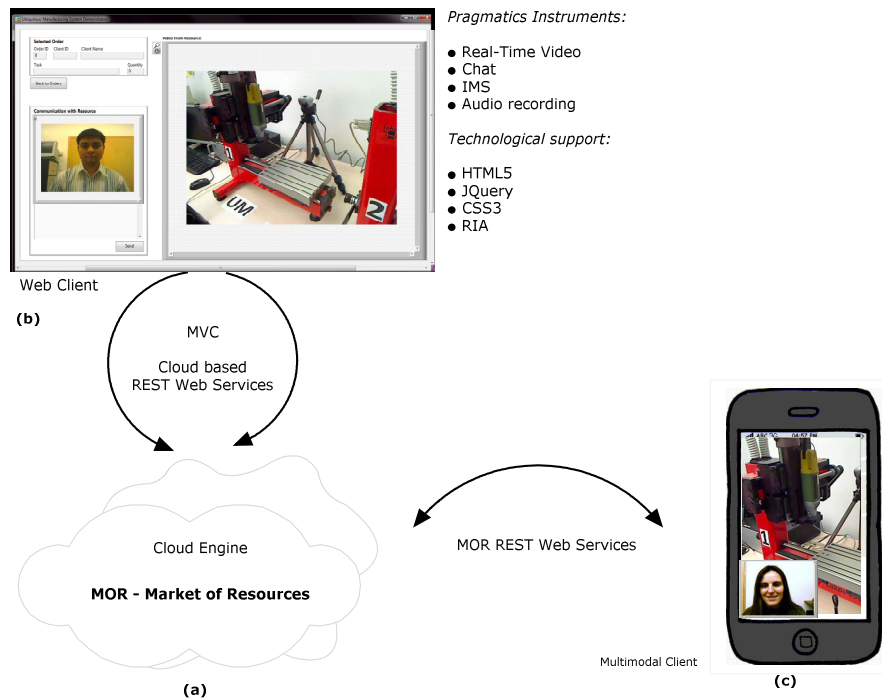


Figure 5.25 - UMS Supporting Architecture

The global Market of Resources will be supported by cloud-based mechanisms (brokering) inherent to SOA Governance. The services, as instances of manufacturing resources (machines, persons, enterprises, etc.), are autonomously maintained following asynchronous subscribing pattern, classified using SLA (Services Layer Agreement) and geo-referenced with spatial data.

The services selection must be agile, sufficiently effective and dynamic to allow advanced search criteria (time, priorities, quality of service, etc.) and react to services asynchronous status notification (free, off, occupied, etc.). Resources' spatial data will provide/help on their localization on map using, for instance, distance, time, costs or facilities criteria. For each resource status and according to specific parameterization, alternative resources must be enabled by brokering service.

If it is not possible to get clear decisions about resources selection, or complementary resource information is needed, direct and synchronous communication must be allowed between stakeholders, so the user experience can be considered.

As a workflow overview, a Process Plan (Figure 5.26 (a)) determines operations and resource specification (like a resource stereotype or meta-resource) (Figure 5.26 (b)) to handle them; broker finds candidates resources able to support it (Figure 5.26 (c)), mapping them to resources

on the ground (Figure 5.26 (d)). The mapping process is not necessarily automatic, but assisted with user participation, if needed, that is, using *Pragmatics* .

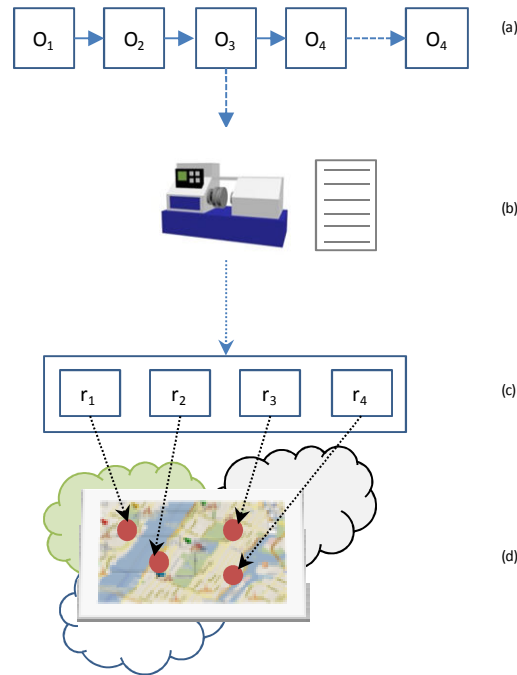


Figure 5.26 - Cloud-based broker: (a) Process Plan (b) Stereotype (c) Candidate resources (d) Spatial Data in cloud

Each resource represents a service (or many) that is hosted on cloud. It has an interface description language (IDL) that allows its discovery, an interoperability specification to follow and an (REST) API that allows its integration with (or, use by) other resources. So, each resource has its own “information system” to handle its work. “Residing” in cloud they are named *cloudlets*.

The application front-end has a RIA Presentation Layer behaving as a *dashboard* that, besides integrating common RIA web components, allows the management of each integrated cloudlet and global monitoring of associated resource (service).

RIA will be supported by emergent web 3.0 technologies (HTML5, JQuery, CSS3, etc.) and pragmatics instruments, communicational channels mainly, due to multimodality requirement, will be supported by open source communication technology as WebRTC and Web Media Capture.

So, each cloudlet is enhanced with layers representing enhanced services. Considering the spatial data representation on google maps, for instance, the map services (Figure 5.27 (a)): zoom, 3D, etc. (supported by google) will be enriched with advanced infowindows (Figure 5.27 (b)) where (existent) communicational channel links will be enable (email, SMS, chat, RT video,

audio recording, etc.) and thus, in dashboard, a direct synchronous conversation with resource's owner will be possible if required (Figure 5.27 (c)).



Figure 5.27 - Cloudlet Architecture (a) Dashboards (b) Cloudlet (service) (c) Enhanced cloudlet (d) Cloudlet with pragmatics instruments

Address a sustainable Interoperability

Interoperability and integration are two distinct and sometimes mistaken terms. Although (IEEE, 1990b) presents interoperability as *the ability of two or more systems or components to exchange information and to use the information that has been exchanged*, they related it with technological compatibility *while sharing the same hardware or software environment*. In another point-of-view, Chen et al. (2008) stress interoperability as the capability to “talk” and integration as “to be part of”. Agreeing with Chen, any system to result integrated, all their “parts” need to be interoperable. In short, interoperability demands coexistence, autonomy and federated environment and integration requires coordination, coherence and standardization.

Considering this, the tentative to get sustainable this interoperability implies more than to get all system components technologically integrated (and interoperable). Besides the cloud architecture (better, computing) supports required infra-structures (IaaS) scalability, flexibility and reliability,

the cloud services (SaaS), developed internally by the enterprise itself or by a third party, need to be useful and so interoperable and composable. An internally service has potential to be externally useful, either solely or composed on a new service (added service). This could happen because the cloud enables efficient mechanisms to discover efficient and appropriated services.

Following the cloud environment, the companies easily share process, data and services, with technological abstraction and independence. This represents an added value for business allowing companies' CIO to focus on their business core, requesting external services for the remaining business part.

Paraphrasing Mullolland (2008), *"If the companies in the supply chain share information through a cloud environment, each participant publishes data to the cloud, leaving the analytics to the cloud service, which can use a search-like approach to provide answers on materials. This is substantially more scalable and cost-effective because it decreases the burden on individual organizations and reduces the overall cost of the solution."*

Overall, SOA patterns (M Endrei, 2004), cloud services and cloud architecture, from its inherent capabilities allow easily discovery of potential (cloud) services and, in consequence, new resultant (added) services promote new capabilities, since they are published in cloud. In addition, the interpretation of the QoS of these services are better supported with pragmatics since allows users (or companies) to analyze contextual and dynamic variables. The criteria to select and compose internal or external resources (services) can be tuned after "learning" with previous experiences or results. The brokering mechanism will dynamically consider these variants, affecting effectiveness ranking criteria, for instance.

Technological failures or inefficiency are easily overcome due to adaptability and reliability of cloud infra-structure. However the same is not true in services (SaaS) level, even SOA and cloud services try to archive that. They try to get semantically matching services but abstract real user perspective. However this is efficiently supported using the innovative communicational feature of our architecture. The brokering allows dynamic reconfiguration of services, departing the "inefficient" and enabling users on direct participation on the co-creation of alternatives.

Being a communicational architecture, users can easily interact in human-to-human model which allows more real alignment with personal point-of-view towards an effective system. Previous success and inefficiency are well shared and interpreted by all participants.

In conclusion, this architecture that supports transactional and communicational interoperability, strengthened with pragmatics support and cloud services reliability (Figure 5.28) is more sustainable since it refers real users and not “abstracted users”, has happened with others pure technological architectures.

	Transactional	Communicational
Multitier	-	+ -
Cloud	+ -	+

Figure 5.28 - Cloud and Communicational architecture enhance sustainable interoperability

6 VALIDATION OF THE COMMUNICATIONAL DIMENSION OF THE PROPOSED ARCHITECTURE

6.1 The proposed architecture validation framework

For the common user, to well understand a concept it must be sufficiently described. However, the information access (web, databases, etc.) requires some intelligence to interpret and to integrate it effectively (Gio, 1992).

We previously saw (Chapter 2) that data heterogeneity arrives from three main categories: its syntax (format), its structure (synonyms, prefix, suffix, etc.) and its semantics (meaning of terms in a particular context). Bu we defend that more categories rely on that heterogeneity, mainly those inherent to personal or subjective interpretation of the context, referred as *Information Field* in Semiotic category.

Considering that:

- there are important solutions to explore syntactic and structural problems, such as the use of mapping between standards like XML, RDF, etc.
- the same happens with semantic integration (Heiner Stuckenschmidt, 2003), as is the case of Ontologies, Terms networks, Thesaurus, Topics Maps, etc.
- the scientific community registers several difficulties on both perspectives and sometimes is need to explicitly describe information semantics (Stuckenschmidt & Harmelen, 2005),

we can conclude that the solution is not only a technological question.

“The attempt to provide interoperability suffers from problems similar to those associated with the communication amongst different information communities. The important difference is that the actors are not persons able to perform abstraction and common sense reasoning about the meaning of terms, but machines. In order to enable machines to understand each other we also have to explicate the context of each system, but on a much higher level of formality in order to make it machine understandable (...)” (Stuckenschmidt & Harmelen, 2005).

Thus we will base our experimentation work on mechanisms to overcome these integration problems towards a non technological (semiotic) integration objective. Collaboration mechanisms (mainly communication) between participants against discordant ontological terms will be the base.

Resuming, the two main dimensions framework to validate the architecture are: a) Technology and b) Communicational capacity (G. D. Putnik *et al.*, 2012). The essence of the experimentation focused the main goal of the research thesis: the relevance of a communicational architecture to achieve effectiveness.

We assume the supporting technology as not critical to get more or less effectiveness. We assured the same technological infra-structure for the experimentation support; we adopted well understood formal mechanisms (ontologies) to describe semantics and to avoid eventual misunderstanding or any kind of technology dependence. However, the set of available services to support direct interaction between participants were the relevant criterion. So, throughout this experimentation we will focus on the use of Pragmatics instruments to deal with these integration problems, neglecting syntactic and semantics integration ones.

This following section describes the experimentation and presents the adopted researching methodology.

6.2 Synopsis

Goal: *Ontologies Interoperability using User Experience and Pragmatics Instruments*

Subject: *Graphical User Interfaces (GUI) description using Ontologies*

Context: GUI in Services Composing

Considerations about ontologies:

- Describe semantically domain terms and their relations
- Allow technology independent modeling
- Add context-ware properties
- Consider target devices and user roles

According to Garret's model (Figure 6.1), the mapping of ontology to the presentation Layer (User Interface) is structured in five layers (Garrett, 2010):

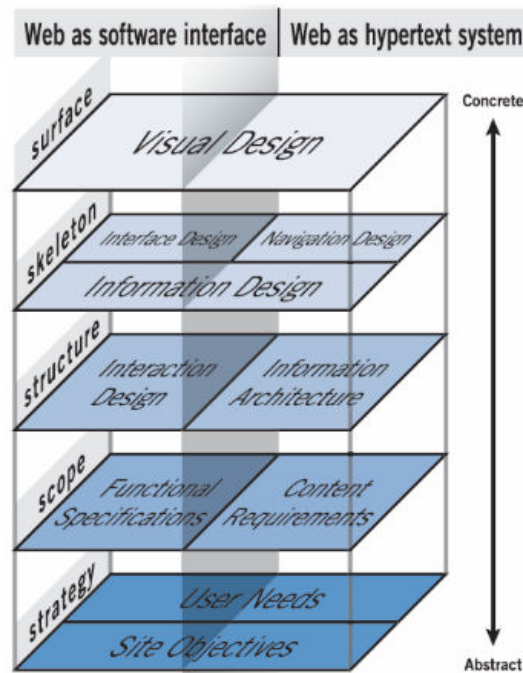


Figure 6.1 - Adapted User Experience Elements

Source: (Garrett, 2010)

Considering the main goal of this experiment, we will give emphasis to the *strategy* (base layer), *scope* and *structure*. The *skeleton* and *surface* layers could be stated in future work.

Strategy: UMS Graphical User Interface!

Scope: Identify domain concepts and sub-concepts (Graphical User Interfaces), vocabulary used. Don't define relationships (structure)!

Structure: Define the logical structure of user interface. Define relationships between concepts and sub-concepts, basically. Provides hierarchical structure and in concepts navigation. In GUI define the way how elements must be arranged and grouped.

6.3 Research Methodology

The main goal of this work focuses ontologies interoperability, contextualized for Graphical User Interfaces (GUI) of services composition. One could note that interoperability problems arrived by technical details, as well as by human and context. We could state these problems on *Podio market app* selection, a pure UI functional detail.

It is known that a human can communicate during learning (Nunes, 2005), and solve their questions or doubts talking with someone else which “*seems able to answer or clarify it*”, we will explore that detail and simulate the same procedure on UI ontologies description.

Since the idea was not to create a new ontology, we follow Garret’s model to map ontologies to the Presentation Layer, considering only *strategy*, *scope* and *structure* layers of his model:

Strategy: User Pragmatics for GUI;

Scope: GUI for Ubiquitous Manufacturing System;

Structure: GUI taxonomy hierarchy.

We defend that three “phenomena” are behind the human-system interaction relation (Figure 6.2). The first, User Interface (UI), comes from the user (human) observation process and describes graphical (ergonomic) and functional details inherent to human as actor of the system, essentially. The second, resulting from the User Experience (UX), enriches the first one with knowledge (validation, sense or judgment) coming from his own experience or context influence, and the third, deriving from User Pragmatics (UX), (possibly) could create a new user interface, since dictates the reasoning and accordance (about initial user interface) on the co-creation (co-design) process.

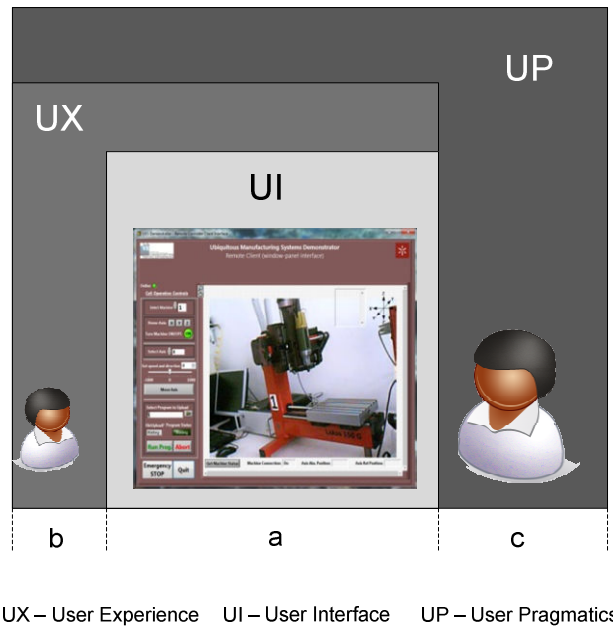


Figure 6.2 - Experimentation stages

To better support this process, we divided the experiment into three main areas of interest (Figure 6.2): A) User Interface ontology (UI), where participants need to create ontologies; B) mapping ontologies with the User Experience (UX), where ontologies ambiguities must be interpreted and validated; and C) mapping ontologies with User Pragmatics (UP), where participants must work together to co-create an interoperable (consensual) ontology.

Thereafter the experiment was structured according to four stages: 1) the creation of an ontology to describe the UI example; 2) the interpretation of created ontology; 3) validation of the achieved interpretation; and finally 4) the co-creation of a correct interpretation ontology.

The experiment started with the creation of an ontology (using Protégé OWL editor) to describe an UI design, corresponding to the presentation layer of an ubiquitous manufacturing system application (area (a) in Figure 6.2). According to his (own) user experience (UX), the created ontology must be interpreted, i.e., the participants must interpret the UI according to their perspective, knowledge and context (area (b) of Figure 6.2). Next, it follows a validation process towards a co-creation of a correct interpretation ontology, using mapping techniques initially and, if needed, pragmatics instruments (mainly conversation), i.e., applying user pragmatics (UP) (stage (c) in Figure 6.2).

Since we want to test the relevance of user pragmatics on effective ontologies interoperability, the key starting point was the following hypothesis:

The problem of Ontologies Interoperability lies in the existence of the Information Field (IF)

With this experimentation we try to achieve details about some assumptions and get information to discuss the theses which support our reasoning. The assumptions were:

1. Ontologies by themselves fail on User Interfaces integration
2. Ontologies *de per si* are not efficient for UI description

And our proposal for the resolution of this problem is the application of Pragmatics instruments, according to next three theses:

Thesis 1: Different IFs exist

Thesis 2: Different IFs influence OI

Thesis 3: Pragmatics is an instrument for the resolution for the OI problem

6.4 Experimentation description

The experimentation will be organized in three main parts:

- Part A - User Interface Description
- Part B - User Experience
- Part C - User Pragmatics

Organized in next 5 steps:

- Step 0: UI Analysis
- Step 1: GUI ontology (GUIO) definition
- Step 2: Interpretation of UIO
- Step 3: Revision of UIO Interpretation
- Step 4: Validation of both Interpretations
- Step 5: Co-creation of new Ontology

Let analyze better each of these steps:

6.4.1 Part A – User Interface Description

Step 0: UI Analysis

Each user must analyze carefully the object of the experiment: an UMS User Interface (Figure 6.3)



Figure 6.3 - UMS UI used in the Experimentation

Step 1: GUI ontology (GUIO) definition

Describe what you see and what you interpret. Mapping domain scope and structure.

Identify *domain (UMS GUI)* and their *concepts* (scope): “text area”, “menu area”, “table”, “area”, “inside”, “aside”, “machine”, “camera view”, “resource”, “attributes”, “volume”, “image”, etc. and *domain concepts relationships* (structure): “area with text and table”, “section with image from machine”, etc.

6.4.2 Part B: User Experience

Step 2: Interpretation of UIO

According to his point of view (experience) each *pair of users* (in each group) interprets previously defined ontologies, noting things like: opinions, ambiguities, agreements, disagreements, unknown, etc.

Step 3: Revision of UIO Interpretation

According to his point of view (experience) each pair of *users* revises the result of previously interpretation (by others revisers) of his initial ontologies, noting things like: opinions, ambiguities, agreements, disagreements, unknown, etc.

6.4.3 Part C: User Pragmatics

Step 4: Validation of both Interpretations

Pair of *users* (both) converse about those previous interpretations and existent ambiguities. It will surely result agreements and disagreements.

Step 5: Co-creation of new Ontology

After previous group discussion and conclusions, members work together in the co-creation of a new ontology, considering agreements and disagreements.

Summary:

Part A		Part B		Part C	
Step 0	Step 1	Step 2	Step 3	Step 4	Step 5
(in class) 45 min		(in class) 30 min	(in class) 30 min	(in class) 60 min	(at home)

6.5 Collaboration 1-to-1

6.5.1 Support

In this experimentation we follow the next conventions:

<i>UI:</i>	User Interface
<i>U_a, U_b:</i>	Users A and B

O_A, O_B : Ontology defined by user A; Ontology defined by user B

$O_A \rightarrow O_B$: Interpretation, or mapping, of Ontology A to Ontology B

$O_{O_A \rightarrow O_B}$: Ontology of interpretation, or mapping, of Ontology A to Ontology B

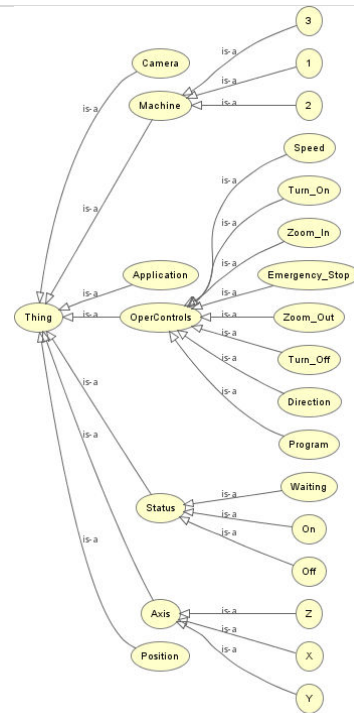
Pre-requisites: GUI analysis; Knowledge about ontologies; Protégé OWL Editor

The experimentation involved: the user interface (UI) (Figure 6.3), the UI designer, here denoted by U_A , the UI user, here denoted by U_B , and the ontology editor (Protégé OWL v4.02). The experiment took 4 hours. The two participants have skills in ontologies, user interfaces specification and software development. Each of them had to follow the next experiment steps.

Step 1. U_A and U_B define their own ontologies to describe the UI

- U_A defined his ontology O_A
- U_B defined his ontology O_B

Result: O_A (Figure 6.4) and O_B (Figure 6.5) for UI



Comment: U_A (UI designer) has defined the ontology O_A to present his design, and U_B (UI user) has defined the ontology O_B to present his perception of the UI. Both ontologies were presented using the OWL editor.

Step 2. Ontology interpretation

U_B interprets (maps) O_A to his own ontology O_B , representing the interpretation, that is, through the interpretation ontology $O_{O_A \rightarrow O_B}$

Result: $O_{O_A \rightarrow O_B}$ informal (Figure 6.6) and formal (Figure 6.7 and Figure 6.8)

Comment: to effectively use the UI, U_B has to interpret correctly the UI designer's ontology to his own ontology, that is, U_B has to map O_A to O_B , formally $O_A \rightarrow O_B$. This interpretation could be formally presented by the interpretation ontology, that is mapping ontology, formally $O_{O_A \rightarrow O_B}$.

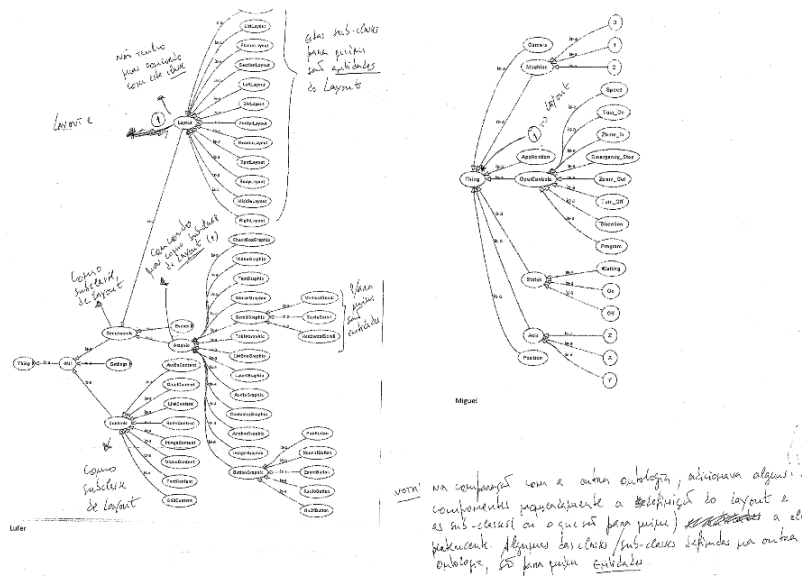


Figure 6.6 - User B's interpretation of O_A

Step 3. Validation of the interpretations

- U_A validates $O_{O_A \rightarrow O_B}$ by U_B

Result: informal description of discrepancies in interpretation ontology (Figure 6.9)

Comment: Some divergence on taxonomy entities was found: *Layout* sub-classes and *ScrollGraphics* sub-classes were interpreted as entities; *Application* class mapped to *GUI*. Ontology structure ambiguities were found too: *Components* and *Contents* were considered *Layout* sub-class, and *Layout* as sub-class of *Thing*.



Figure 6.7- User B's interpretation of O_A – formal graphical

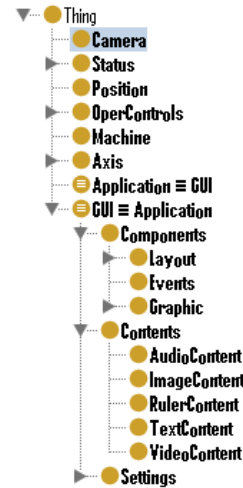


Figure 6.8 - User B's interpretation of O_A – formal textual

The divergences imply that the two ontologies, O_A and O_B , are virtually not interoperable as expected, that is, U_B will not be capable to effectively and efficiently use the UI, i.e., there is a problem of O_A and O_B interoperability. The cause of these divergences, that is, of the ontologies interoperability problem, is in fact, in the different information fields of U_A and U_B .

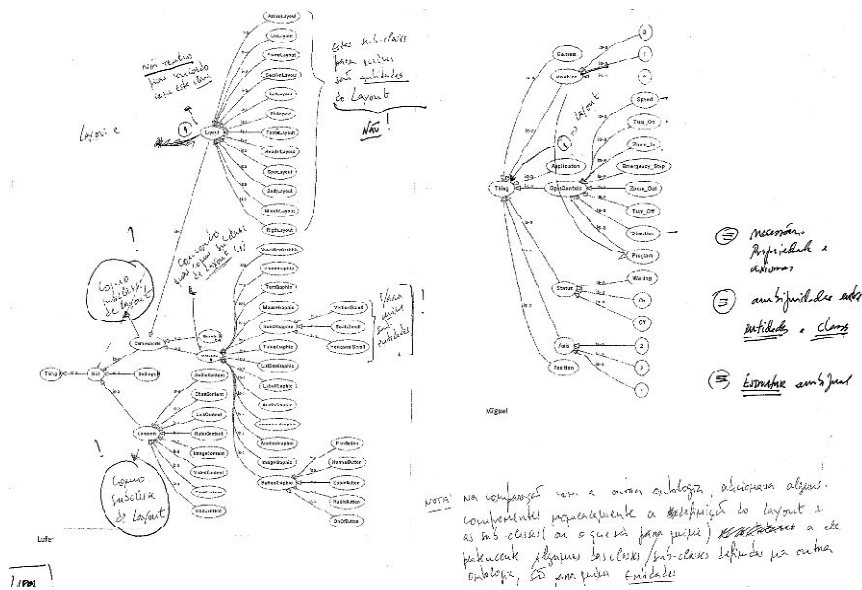


Figure 6.9 - U's validation of interpretation ontology by U.

Step 4. Co-creation of correct interpretation ontology

- a. U_A and U_B converse about interpretation of $O_{O_A \rightarrow O_B}$ made by U_B and construct the correct interpretation $O^C_{O_A \rightarrow O_B}$

Result: Correct interpretation ontology $O^C_{O_A \rightarrow O_B}$ formal textual (Figure 6.10) and formal graphical (Figure 6.11)

Comment: The correct interpretation ontology from O_A to O_B , formally $O^C_{O_A \rightarrow O_B}$, is a condition for effective and efficient ontology interoperability. The correct interpretation ontology can result only from human communicational process, which involves pragmatics aspects, as pragmatics, by definition, refers to concrete and individual users – while semantics abstracts the concrete individual users. Further it implies that the interpretation ontology $O_{O_A \rightarrow O_B}$ is not valid for interoperability with some third user U_C which, in principle, has his own ontology O_C of the same UI, and a new co-creation of new interoperability ontology is necessary.

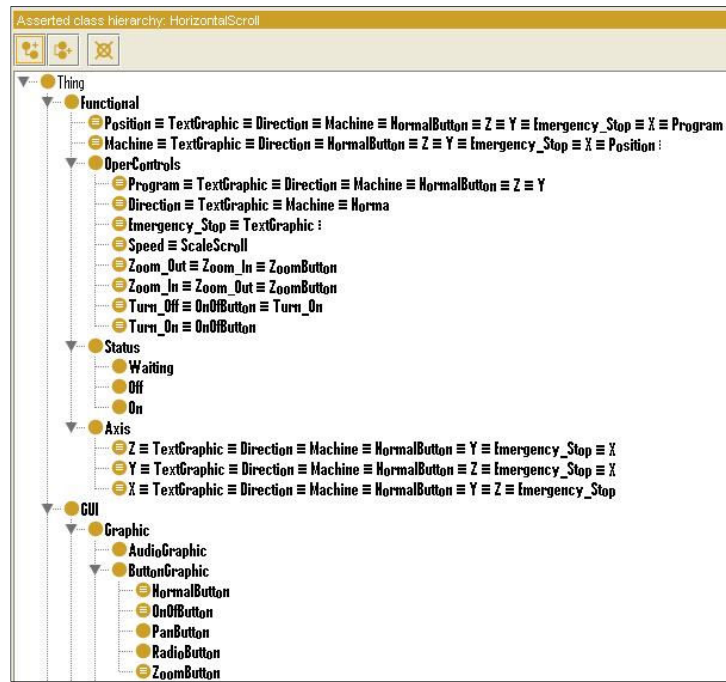


Figure 6.10 - Excerpt of co-created correct interpretation ontology – formal textual representation

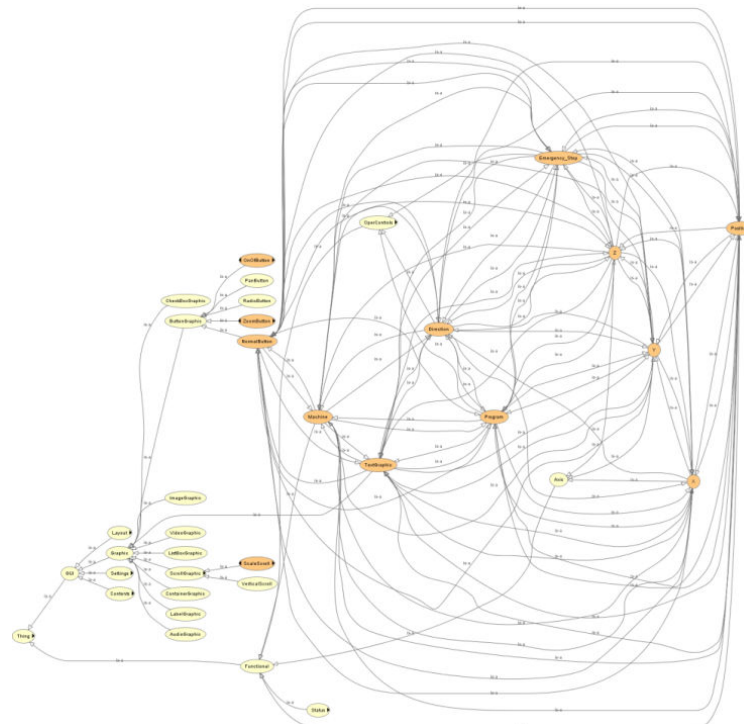


Figure 6.11 - Co-created correct interpretation ontology– formal graphical representation

6.5.2 Findings

Analyzing the interpretation ontologies, that is, the differences between the $O_{O_A \rightarrow O_B}$ and $O_{O_A \rightarrow O_B}^C$, it could be identified the existence of two group of mappings of the $O_{O_A \rightarrow O_B}^C$.

The first group represents the mappings between the classes which were considered equivalent before the co-creation process (step 2 and 3) and which were present in $O_{O_A \rightarrow O_B}$. This group represents, in fact, the common part of the two IF of U_A and U_B (Figure X – a), and which is represented by white area in Table 6.1.

The second group represents the mappings between the classes which were considered equivalent after co-creation process (step 4), that is, the second group represents the result of the co-creation process, representing, in fact, the difference between $O_{O_A \rightarrow O_B}$ and $O_{O_A \rightarrow O_B}^C$.

Further, adding this second group to the first group of mappings means the unification of two initially different Information Fields, (Figure X – b), and which is represented by the grey area in Table 6.1. For instance: class *Application* of O_B was mapped with class *GUI* of O_A , and classes *Position* and *Machine* of O_B were mapped to *TextGraphic* of O_A ; unmapped class *Components* of O_B and *Camera* of O_A “disappeared” (see the grey area of Table 6.1).

Full and effective interoperability between ontologies is just an interpretation of the information fields unification. This unification is possible only through the human communicational process.

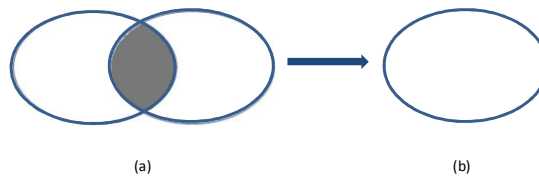


Figure 6.12 - (a) Two Information Fields or ontologies with a common part, and (b) Unified Information Field or ontology after pragmatics, co-creation, process

Looking back the proposed three thesis: *Thesis 1*: Different Information Fields (IF) exist; *Thesis 2*: Different IFs influence OI; and *Thesis 3*: Pragmatics is an instrument for the resolution for the OI problem, and considering the global experiment results we have demonstrated that all these are confirmed in different the experiment steps, as follows:

a) the results of *step 2* prove *Thesis 1*, since different users had different interpretation for the same object (UI) made in *step 1*, i.e., proves the existence of different Information Fields;

		<i>NormalButton</i>	<i>OnOffButton</i>	<i>ZoomButton</i>	<i>TextGraphic</i>	<i>GUI</i>	<i>Components</i>
<i>Status</i>	<i>Waiting</i>	✓					
	<i>On</i>	✓					
	<i>Off</i>	✓					
<i>Axis</i>	<i>Z</i>	✓					
	<i>X</i>	✓					
	<i>Y</i>	✓					
<i>OperControl</i>	<i>Direction</i>	✓					
	<i>Emergency-stop</i>	✓					
	<i>Zoom</i>			✓			
	<i>Turn</i>		✓				
	<i>Application</i>					✓	
	<i>Position</i>				✓		
	<i>Machine</i>				✓		
	<i>Camera</i>						

Table 6.1 - Some mappings and ambiguities

b) the results of *step 3* prove *Thesis 2*, since, different Information Fields (and inherent different interpretations) made the immediate ontology interoperability impossible; and

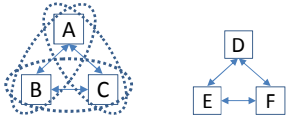
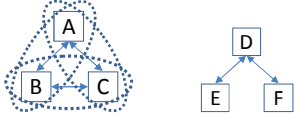
c) the results of *step 4* prove *Thesis 3*, since, after the co-creation process that involved pragmatics aspects (using conversation as pragmatics instrument), they accorded and co-created a correct interpretation ontology, assuring the effective individual ontologies interoperability.

Referring the experiment results and their representation by Table 6.1 to the concepts of UI, UX and UP (Figure 6.2), it could be said that the “white area” of Table 6.1 corresponds to UI+UX, while the “grey area” of Table 6.1 corresponds to UP. In this way, the pertinence and validity of the UP concepts, introduced by the authors, is demonstrated.

6.6 Collaboration N-to-N

6.6.1 Dynamics

	12 participants (students)											
Steps	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 2px 5px;">A</div> <div style="border: 1px solid black; padding: 2px 5px;">B</div> <div style="border: 1px solid black; padding: 2px 5px;">C</div> <div style="border: 1px solid black; padding: 2px 5px;">D</div> <div style="border: 1px solid black; padding: 2px 5px;">E</div> <div style="border: 1px solid black; padding: 2px 5px;">F</div> <div style="border: 1px solid black; padding: 2px 5px;">G</div> <div style="border: 1px solid black; padding: 2px 5px;">H</div> <div style="border: 1px solid black; padding: 2px 5px;">I</div> <div style="border: 1px solid black; padding: 2px 5px;">J</div> <div style="border: 1px solid black; padding: 2px 5px;">K</div> <div style="border: 1px solid black; padding: 2px 5px;">L</div> </div> <div style="display: flex; justify-content: space-around; margin-top: 5px;"> G1G2G3G4 </div>											
S ₁	<i>Criation of Ontologies</i>											

	$O_A O_B O_C \dots O_L$
S_2	<i>Ontologies interpretation</i>
	$A \rightarrow B \mid A \rightarrow C \mid B \rightarrow C \mid B \rightarrow A \mid C \rightarrow B \mid C \rightarrow A \mid I \rightarrow J \mid J \rightarrow I$ $K \rightarrow L \mid L \rightarrow K$ $D \rightarrow E \mid D \rightarrow F \mid E \rightarrow D \mid E \rightarrow F \mid F \rightarrow D \mid F \rightarrow E$ $G \rightarrow H \mid H \rightarrow G$
S_3	<i>Revision of Ontologies interpretation</i>
	 $(A \rightarrow B \rightarrow C) \mid (D \rightarrow E \rightarrow F) \mid G \rightarrow H \rightarrow I \mid J \rightarrow K \rightarrow L$
$S_4 + S_5$	<i>Revision Validation + Ontology creation</i>
	 $O_{ABC} \quad O_{DEF} \quad O_{GHI} \quad O_{JKLI}$

6.6.2 Monitorization

Initially were two hypotheses to monitor the experimentation

- H1: All interpretations are made in written text directly in the documents where ontologies were printed
- H2: Will be use specific tables (attached in appendix)

We decide for hypothesis H2. We created a specific table (**Table A**) to register all interpretations.

Documents to monitor the interpretation (Steps 2, 3 and 4)

- In step 1 the author creates his ontology in Portege SW, and prints the ontology diagram in PDF.

- In step 2 – *Interpretation of ontologies*, each author, when interpreting the ontology of another, uses the **Table A**, and notes in column 1 the terms on which has an opinion! It must use the values of the existing agreement scale as well as the justification for such value (in column 2).
- In step 3 – *Review of the interpretations*, each author, alone, reviews the interpretation that was made to his ontology. Using the column 3 of the **Table A**, expresses his perspective using the existing agreement scale.
- In step 4 – both authors converse about the interpretations and revisions. Both use the column 4 of the **Table A** to record their findings.
- In step 5 – both authors try to come up with a new ontology that "meets" the prospect of both.

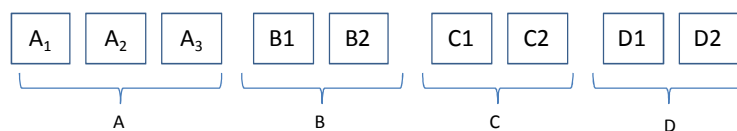
Agreement Scale:

- ❶ - Totally agree
- ❷ - Agree
- ❸ - Disagree
- ❹ - Completely disagree
- ❺ - I Do Not Know

6.6.3 Development

The experimentation involved 9 students, organized in 4 groups. One group with three students (Group A) and three groups with two students (Groups B, C and D). Since the number of students were not sufficient to support all groups, one student were member of group A and later also member of group D.

Participants



Group A	Group B	Group C	Group D
A1 – Helder A2 – Vitor A3 – Liliana	B1 – Marcelino B2 – Susana	C1 – Ricardo C2 – Tiago	D1 – Carlos D2 – Liliana (participant A3)

6.6.4 Step 1 – Taxonomies

The number of terms of taxonomies resultant from Step 1 is displayed in Table 6.2. The total of terms presented includes also those terms that are repeated between ontologies. Easily come across that the quantity of used terms was different, thus implies also that ontologies were different.

For example, A2 used 36 terms in his taxonomy while A1 needed just 16. The runtimes of this phase were very close. However, the results do not allow deducing a direct relationship between the number of terms of the ontology and the time taken for the set. Note, for example, in the minimum of time difference used in the ontology of A1 (with only 16 terms) with the A2 (with 36 terms, more than doubled from A1).

	<i>Group</i>	<i>Taxonomy</i>	<i>%</i>	<i>Time (min)</i>
A1	A	16	9%	38
A2		36	19%	37
A3		20	11%	41
B1	B	22	12%	43
B2		25	13%	41
C1	C	24	13%	45
C2		21	11%	36
D1	D	23	12%	36
Total		187		
Average		23,375		
Standard Deviation		5,8		

Table 6.2 - Step 1 resultant Taxonomies

These small differences can be understood, not just necessarily because participants had very different profiles (although they are in fact), but because the way they interpreted the interface was clearly different.

Below are the eight resultant ontologies.

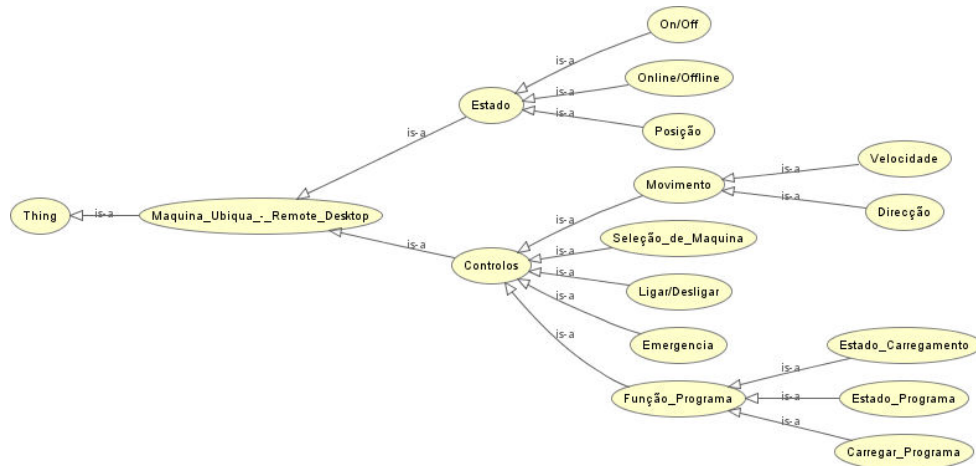


Figure 6.13 - A1 Ontology

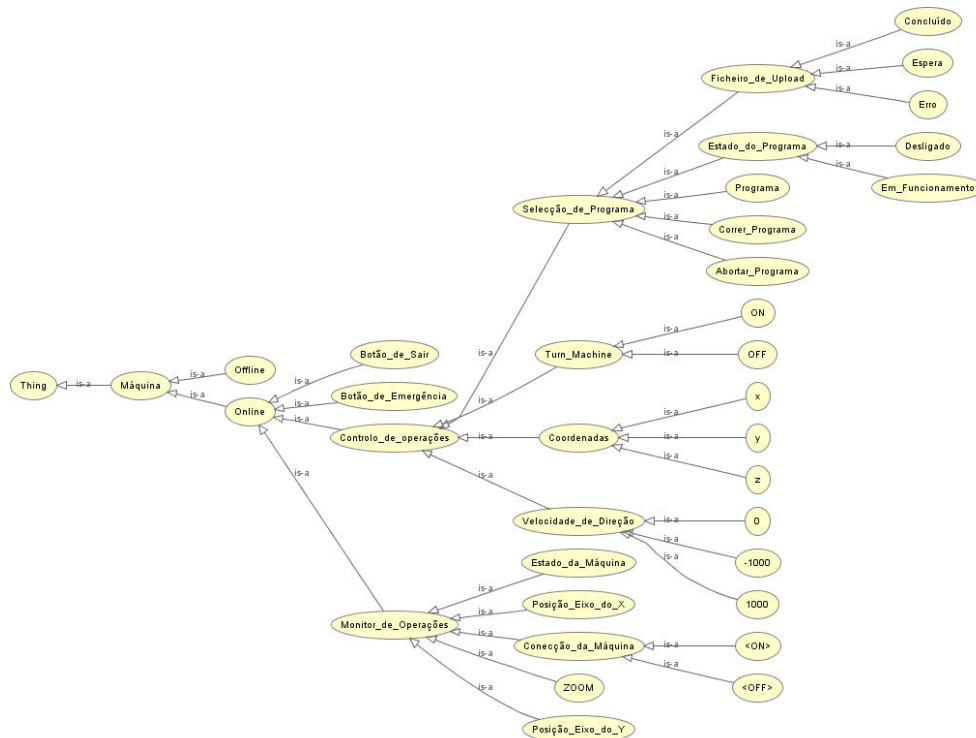


Figure 6.14 - A2 Ontology



Figure 6.15 - A3 Ontology



Figure 6.16 - B1 Ontology



Figure 6.17 - B2 Ontology



Figure 6.18 - C1 Ontology

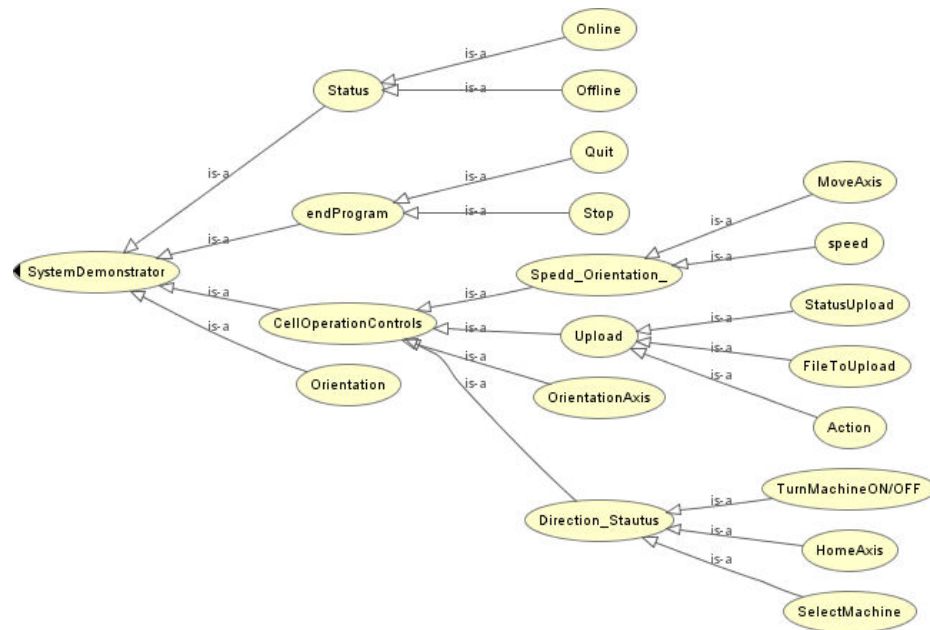


Figure 6.19 - C2 Ontology

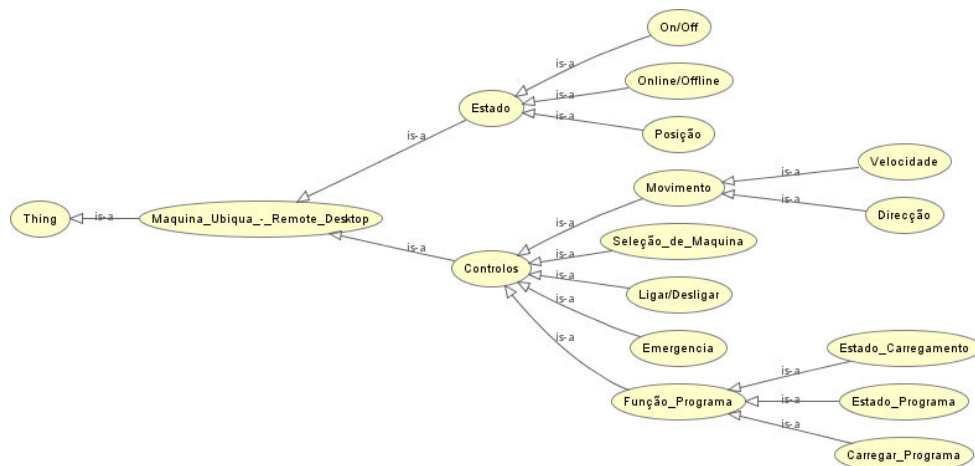


Figure 6.20 - D1 Ontology

6.6.5 Step 2 – Ontologies Interpretation

During this stage of the experimentation, each member of the group was able to express in writing his level of agreement - using a range of values from 1 to 5 of agreement (Figure 6.21) - with regard to the terms used in the analyzed ontology. The interpretation results essentially and naturally from each one's experience – *User Experience* – and was focused in the designation

used in different terms as well as their hierarchical location in the taxonomic used in each ontology.

Grupo:	Autor da Ontologia:	❶ - Concordo totalmente ❷ - Concordo ❸ - Discordo ❹ - Discordo completamente ❺ - Ignoro				
		PASSO 2	PASSO 3		PASSO 4	
Taxonomia	Interpretação + Justificação ❶❷❸❹❺		Revisão da Interpretação ❶❷❸❹❺		Análise Conjunta → Nova Ontologia	

Figure 6.21 - Agreement Scale

Table 6.3 summarizes the analysis work performed by each member of the group during this step. By analyzing the data, the agreements (resulting from the union of the values 1 and 2 of the agreement scale) and disagreements (resulting from the union of the values 3, 4 and 5 of the agreement scale) there represented are significant, reflecting discrepancies in how the ontologies were created and then interpreted.

Another important consideration is the time factor. The time used in this step of the experimentation is not directly proportional to the amount of terms we have to interpret in the ontology. On average, the times are very close but, as noted earlier, concerning the number of used terms, the ontologies were very distinct.

We can then conclude that while formalizing used with ontologies has allowed developing a common and understandable way to describe the subject under study, the result translates into a distinct set of ontologies.

Reviewer	Author	Disagree	Agree	Time (min)
Marcelino	Susana	13	8	27
Susana	Marcelino	5	3	21
Susana	Ricardo	7	0	26
Ricardo	Susana	5	0	26
Marcelino	Tiago	4	4	24
Tiago	Marcelino	2	10	23
Tiago	Ricardo	3	3	18
Ricardo	Tiago	4	2	25
Carlos	Liliana	0	4	18

Liliana	Carlos	8	2	22
Liliana	Helder	3	2	23
Helder	Liliana	8	2	28
Vitor	Liliana	4	0	18
Liliana	Vitor	7	0	28
Vitor	Helder	3	2	19
Helder	Vitor	4	4	24

Table 6.3 - Step 2 Agreements summary

6.6.6 Step 3 – Revision of the interpretation

During this phase, as presented in the description of the experimentation, each user should express an opinion about the interpretation and agreement to its ontology, made by the other member of the group. That is, a process that allows the author to accept or not the interpretation that the other did on his ontology. Table 6.4 summarizes the result of this work.

As can be seen, when comparing with the results obtained after step 2 (Table 6.3), the numbers for disagreements are significantly lower.

<i>Reviewer</i>	<i>Author</i>	<i>Disagree</i>	<i>Agree</i>	<i>Time (min)</i>
Marcelio	Susana	2	1	23
Susana	Marcelino	1	4	24
Susana	Ricardo	1	2	25
Ricardo	Susana	4	2	27
Marcelino	Tiago	3	2	27
Tiago	Marcelino	0	2	15
Tiago	Ricardo	0	2	15
Ricardo	Tiago	2	4	24
Carlos	Liliana	1	3	21

Liliana	Carlos	3	1	20
Liliana	Helder	1	2	25
Helder	Liliana	5	1	20
Vitor	Liliana	1	4	23
Liliana	Vitor	1	3	23
Vitor	Helder	0	8	23
Helder	Vitor	1	4	18

Table 6.4 - Step 3 summary

6.6.7 Step 2 and Step 3 – Mapping of Concepts

After the work of interpretation, alone and in group, of all peer members, achieved during steps 2 and 3 of the experimentation, analyzing the reasoning or justification of agreement expressed by all, whether interpreting with participants what they agreed or disagreed, it was possible to identify and present the set of terms mapped between peers.

The tables used to present these mappings are arrays where the symbol ✓ indicates the mapping between the term of the current line and column. In the columns are the terms of ontologies mapped from one of the participants, being rows for the terms of the ontology mapped from the other. The unmapped terms are not presented.

At the end of each table are the total of disagreements and agreements. Let's see then each one of the results between peers.

A1↔A2

	Máquina	Controlos_de_Operações	Selecao_do_Programa	Ficheiro_de_Upload	Velocidade_de_Direcção	Botao_de_Emergencia	Conexão_da_Máquina	ON	OFF
Maquina_Ubiqua_Remote_Desktop	✓								
Estado	✓								
Controlos		✓							
Funcao_Programa			✓						
Carregar_Programa				✓					
Velocidade					✓				
Emergencia						✓			
Ligar/Desligar							✓		
ON/OFF								✓	✓
Summary:	Desagreed: 7				Agreed: 6				

Table 6.5 - A1↔A2 Mapping

From the total number of terms involved in the ontologies created by A1 and A2 (52 terms - Figure 6.13 and Figure 6.14), only 34.6% of the concepts were mapped (Chart 6.1), i.e., understood equivalent by both.

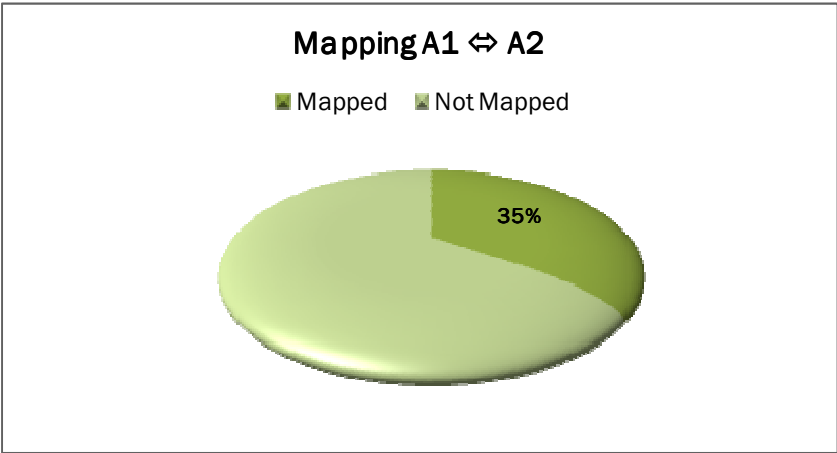


Chart 6.1 - A1↔A2 Mapping results

A2 ⇔ A3

	<i>Controlos_de_Operações</i>	<i>Ficheiro_de_Upload</i>	<i>Posicao_eixo_X</i>	<i>Posicao_eixo_Y</i>	<i>Coordenadas</i>	<i>Velocidade_de_Direcção</i>	<i>Estado_da_Máquina</i>	<i>Conexão_da_Máquina</i>
<i>operation_controls</i>	✓							
<i>file_upload</i>		✓						
<i>axis_position</i>			✓	✓				
<i>Axis</i>					✓			
<i>Speed</i>						✓		
<i>Machine_Conexion</i>								✓
<i>turn_machine</i>							✓	
<i>Summary:</i>	Desagreed: 11				Agreed: 0			

Table 6.6 - A2 ⇔ A3 Mapping

From the total of the terms involved in the ontologies created by A2 and A3 (56 terms - Figure 6.14 and Figure 6.15), only 25% of the concepts were mapped (Table 6.6, Chart 6.2).

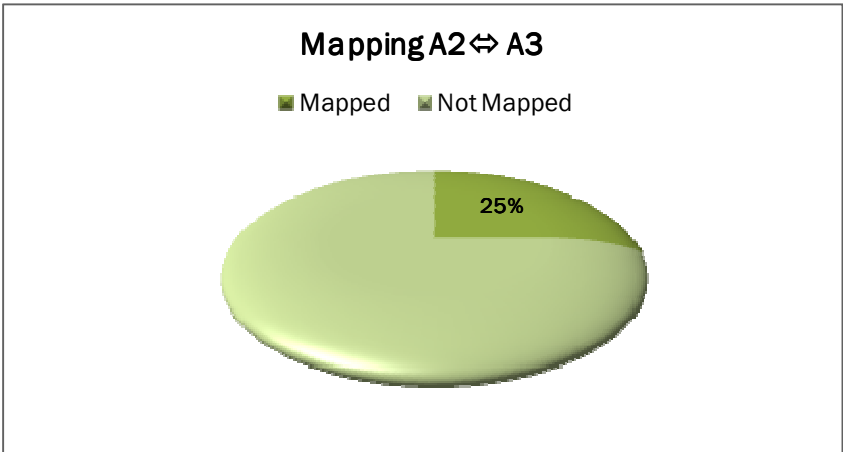


Chart 6.2 - A2 ⇔ A3 Mapping Results

A1 ↔ A3

	<i>operation_controls</i>	<i>axis_position</i>	<i>turn_machine</i>	<i>speed</i>	<i>file_upload</i>
<i>Controlos</i>	✓				
<i>Posicao</i>		✓			
<i>Seleção_de_Maquina</i>			✓		
<i>Velocidade</i>				✓	
<i>Carregar_Programa</i>					✓
<i>Ligar/desligar</i>			✓		
<i>Summary:</i>	Desagreed: 11		Agreed: 4		

Table 6.7 - A1 ↔ A3 Mapping

From the total of the terms involved in the ontologies created by A1 and A3 (36 terms - Figure 6.13 and Figure 6.15), only 30.5% of the concepts were mapped (Table 6.7, Chart 6.3).

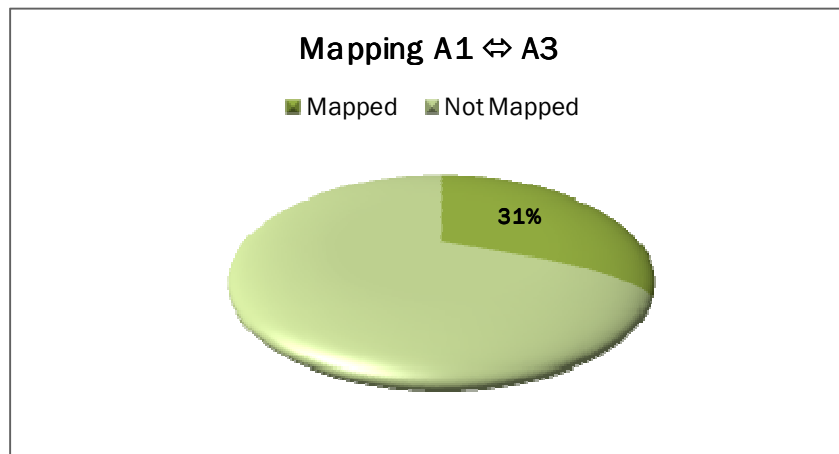


Chart 6.3 - A1 ↔ A3 Mapping Results

B1↔B2

	UploadProgram	Program Status	Direction	NameMachine	MachineConnection	MachineStatus	X	Y	Z
Programa	✓								
Estado_Programa		✓							
Direccao			✓						
Posicao_relativa_X							✓		
Posicao_relativa_Y								✓	✓
Posicao_relativa_Z									
Posicao_absoluta_X							✓		
Posicao_absoluta_Y								✓	
Posicao_absoluta_Z									✓
Num_Maquina				✓					
Conexao					✓				
Estado_da_Maquina						✓			
Summary:	Desagreed: 18				Agreed: 11				

Table 6.8 - B1↔B2 Mapping

From the total of the terms involved in the ontologies created by B1 and B2 (47 terms - Figure 6.16 and Figure 6.17), only 44.7% of the concepts were mapped (Table 6.8, Chart 6.4).

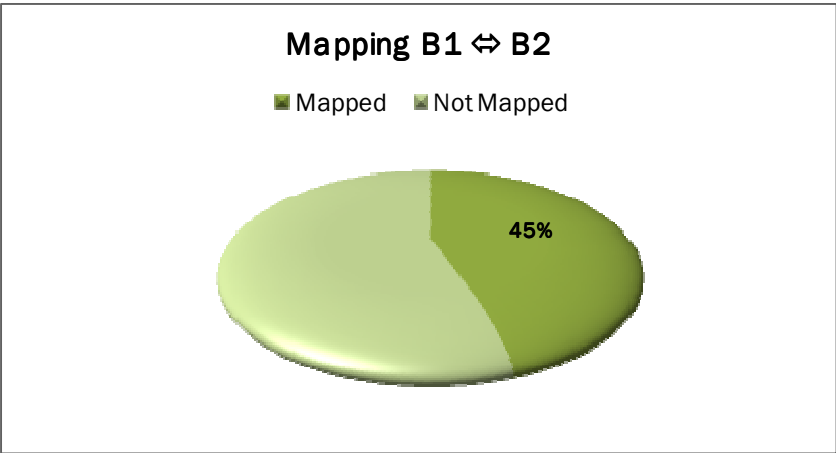


Chart 6.4 - B1↔B2 Mapping

C1 ↔ C2

	Operation_Controls	Set_Speed_and_Direction	Select_Machine	Turn_Machine_On_Off	Home_Axis	Op_Axis
CellOperationControls	✓					
Speed_Orientation		✓				
Direction_Status		✓				
SelectMachine			✓			
Turn_MachineON/OFF				✓		
Status				✓		
HomeAxis					✓	
Orientation_Axis						✓
Axis						✓
Direction_Status				✓		✓
Summary:	Desagreed: 7			Agreed: 5		

Table 6.9 - C1 ↔ C2 Mapping

From the total of the terms involved in the ontologies created by C1 and C2 (45 terms - Figure 6.19 and Figure 6.18), only 48.9% of the concepts were mapped (Table 6.9, Chart 6.5).

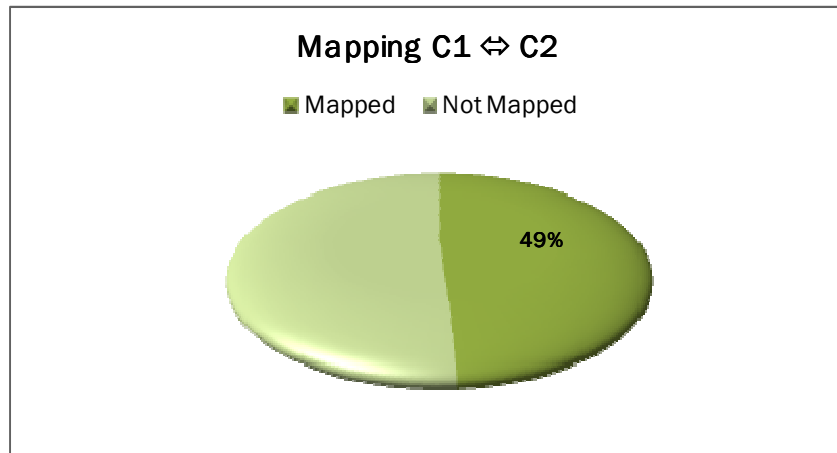


Chart 6.5 - C1 ↔ C2 Mapping

D1 ⇔ D2

	Axis	Turn_Machine	Speed	File_Upload	X	Y	Z
Axis	✓						
Machine On		✓					
Machine Off		✓					
Home_X					✓		
Home_Y						✓	
Home_Z							✓
Change_Speed_X			✓				
Change_Speed_Y			✓				
Change_Speed_Z			✓				
Program				✓			
Summary:	Desagreed: 8			Agreed: 6			

Table 6.10 - D1 ⇔ D2 Mapping

From the total of the terms involved in the ontologies created by D1 and D2 (43 terms - Figure 6.20 and Figure 6.15), only 39.5% of the concepts were mapped (Table 6.10, Chart 6.6).

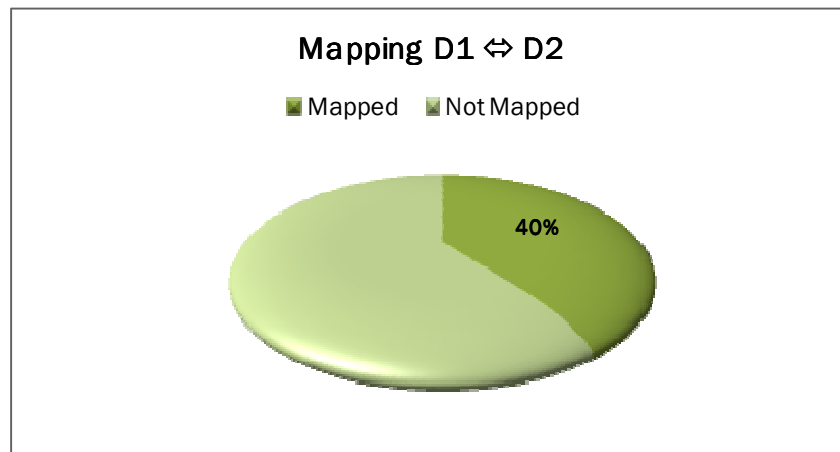


Chart 6.6 - D1 ⇔ D2 Mapping

B1 ↔ C2

	Status	OnLine	OffLine	Estado_do_Programa	Velocidade	Direcao	Eixo	Programa	EstadoPrograma
Estado	✓								
Estado-da-Conexao		✓	✓						
Estado-da-Maquina		✓	✓						
EndProgram				✓					
Quit				✓					
Stop				✓					
Speed					✓				
SpeedOrientation						✓			
OrientationAxis							✓		
Orientation						✓			
MoveAxis							✓		
Upload								✓	
StatusUpload									✓
Summary:	Desagreed: 6				Agreed: 14				

Table 6.11 - B1 ↔ C2 Mapping

From the total of the terms involved in the ontologies created by B1 and C2 (43 terms - Figure 6.16 and Figure 6.19), only 37.2% of the concepts were mapped (Table 6.11, Chart 6.7);

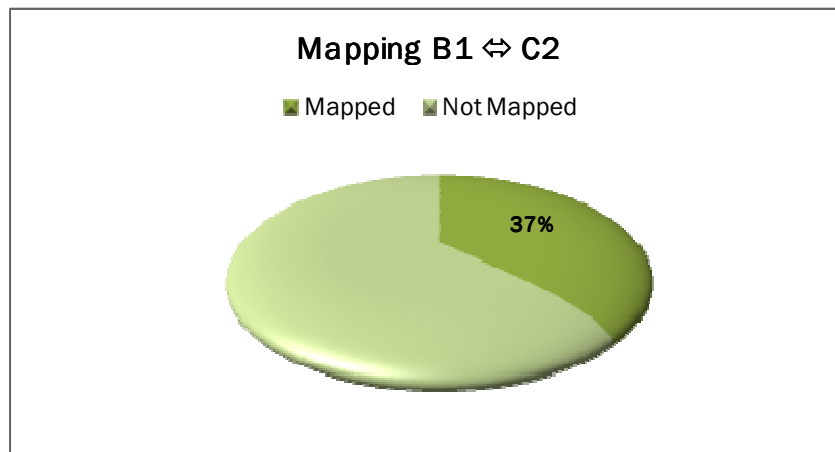


Chart 6.7 - B1 ↔ C2 Mapping

B2 ⇔ C1

	ControlOperation	MoveAxis	HomeAxis	NameMachine	UploadProgram	MachineStatus	Offline	Online	Direction	Speed	X	Y	Z
Operation_Controls	✓												
OP_Axis		✓											
Home_Axis			✓										
Select_Machine				✓									
Select_Program_to_Upload					✓								
Get_Machine_Status						✓							
Turn_Machine_ON_OFF							✓	✓					
Select_Axis											✓	✓	✓
Set_Speed_and_Direction									✓	✓			
Summary:	Desagreed: 12						Agreed: 0						

Table 6.12 - B2 ⇔ C1 Mapping

From the total of the terms involved in the ontologies created by B2 and C1 (49 terms - Figure 6.17 and Figure 6.18), only 22.5% of the concepts were mapped (Table 6.12, Chart 6.8);

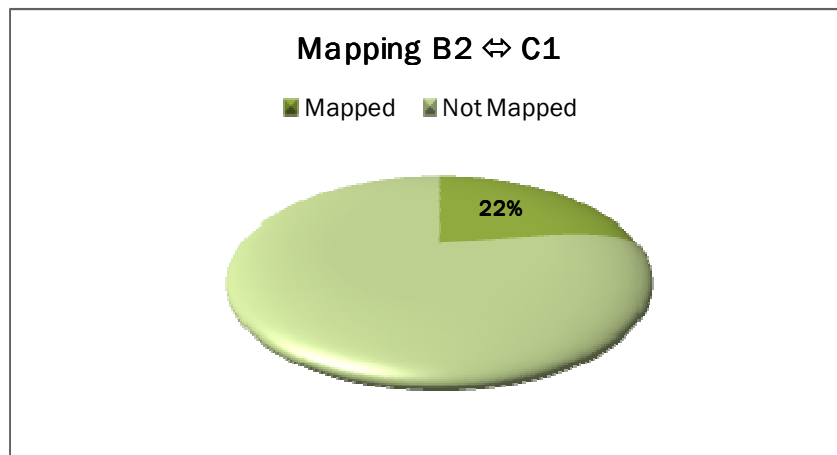


Chart 6.8 - B2 ⇔ C1 Mapping

In short, from the analysis of each own interpretations, there is a great percentage of terms considered appropriate or not applicable (Table 6.13).

Peer	Mapping
A1-A2	35%
A2-A3	25%
A1-A3	31%

B1-B2	45%
B2-C1	22%
C1-C2	49%
B1-C2	37%
D1-D2	40%

Table 6.13 - Mapping Percentages Summary

With this mapping was possible to prepare the next step of the experimentation. In that step is intended that participants will jointly examine the achieved mapping, identify the need for new elements or the possibility to ignore or remove elements.

6.6.8 Step 4 – Joint analysis of ontologies

This stage of the experimentation requires that each pair of participants worked together in the analysis of the interpretations made by both on their own ontologies. Both expressed agreements or differences in interpretations, and worked together in person for the definition of new taxonomies to sustain a new ontology that satisfy both.

Table 6.14 shows the results obtained with this step. It is easy to see that the number of disagreements has declined radically

We can deduce that, after this "conversation" to exchange opinions between the different participants, the explanation and argumentation of the disagreements expressed in previous processes were sufficiently able to put both in agreement on most of the not agreed concepts.

If there is agreement on the differences with regard to taxonomies used by each user, the conditions for the co-creation of a new taxonomy and a new ontology that satisfies both the participants are ensured. Step 5 of the experimentation was responsible for this.

<i>Author</i>	<i>Auhtor</i>	<i>Disagree</i>	<i>Agree</i>	<i>Time (min)</i>
Marcelio	Susana		2	24
Susana	Marcelino		5	
Susana	Ricardo		3	27
Ricardo	Susana	3	3	

Marcelino	Tiago	4	24
Tiago	Marcelino	4	
Tiago	Ricardo	2	25,5
Ricardo	Tiago	2	
Carlos	Liliana	4	24
Liliana	Carlos	4	
Liliana	Helder	6	22,5
Helder	Liliana	3	
Vitor	Liliana	5	21
Liliana	Vitor	4	
Vitor	Helder	8	19,5
Helder	Vitor	5	

Table 6.14 - Step 4 Summary

6.6.9 Step 5 – Co-Creation of the new Ontology

It was asked now that, following the joint analysis of the terms differences and similarities, and from the resultant mapping of terms, participants develop a taxonomy and apply it in a new ontology to describe the subject of the study so acceptable to both. Then appeared new terms as well as there were terms that were eliminated.

All decisions were taken in working together and physically present, with the direct participation and collaboration between all members of the group.

In this last step of the experimentation, Group A, being composed by three elements, worked together and also co-created an ontology.

Below are the most significant transformations in the new ontologies.

Concepts		
New	Mapped	Removed
<p>Maiores_0_e_menor_1000</p> <p>Menores_0_e_maiores_1000</p>		<p>On-line,</p> <p>Off-Line Speed,</p> <p>Machine_connection,</p> <p>select_machine,</p> <p>Estado,</p> <p>Posicao,</p> <p>Movimento Desligado,</p> <p>Em_Funcionamento</p>
Total of terms: 27		

Table 6.15 - A1 ↔ A2 ↔ A3 Co-Creation

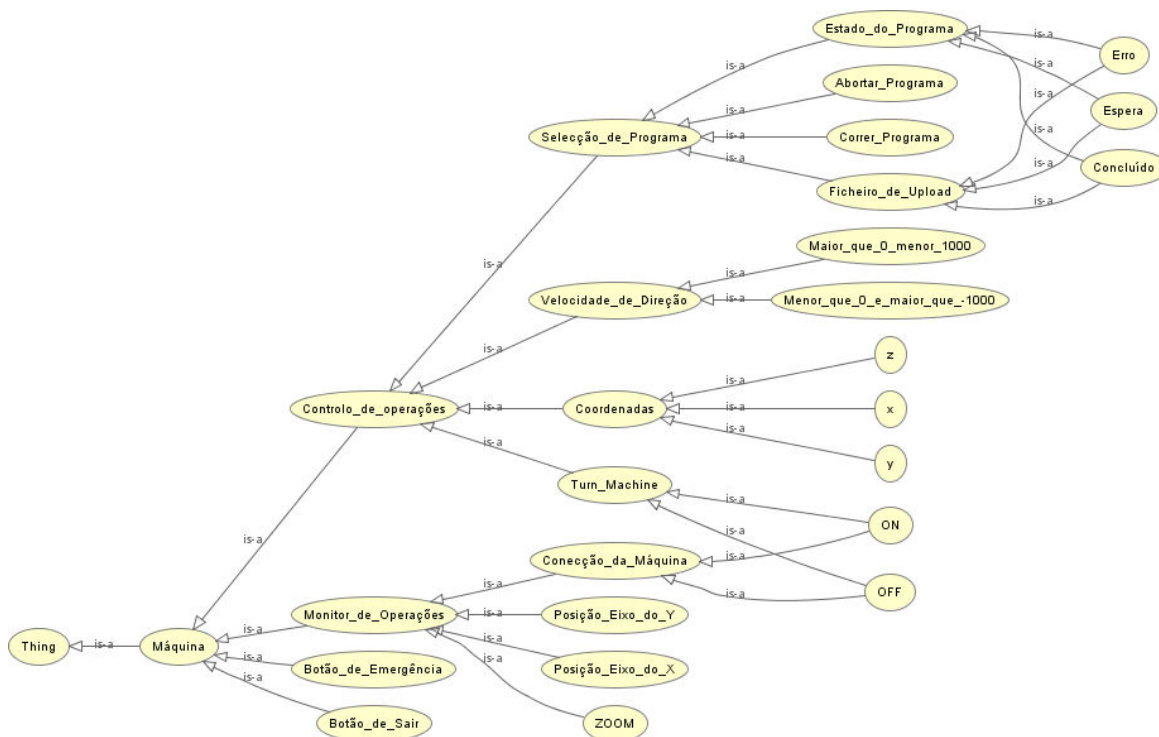


Figure 6.22 - Co-created Ontology between A1 ↔ A2 ↔ A3

Concepts		
New	Mapped	Removed
Axis	Eixo ⇔ MoveAxis [XYZ] ⇔ Axis_[XYZ]_Position Posicao_Eixo_[XYZ] ⇔ Axis_[XYZ]_Position Posicao_relativa_[XYZ] ⇔ Axis_[XYZ]_Rel_Position Posicao_absoluta_[XYZ] ⇔ Axis_[XYZ]_Abs_Position Direcao ⇔ Direction Estado_Programa ⇔ ProgramStatus	Quit, Offline, Online ON, Off, Abort RunProgram, Waiting Num_Maquina, Velocidade, Estado, Estado_da_conexao, Estado_da_maquina, Conexao, Peca, Peca_Mecanica,Peca_electronica
Total of terms: 22		

Table 6.16 - B1⇔B2 Co-Creation

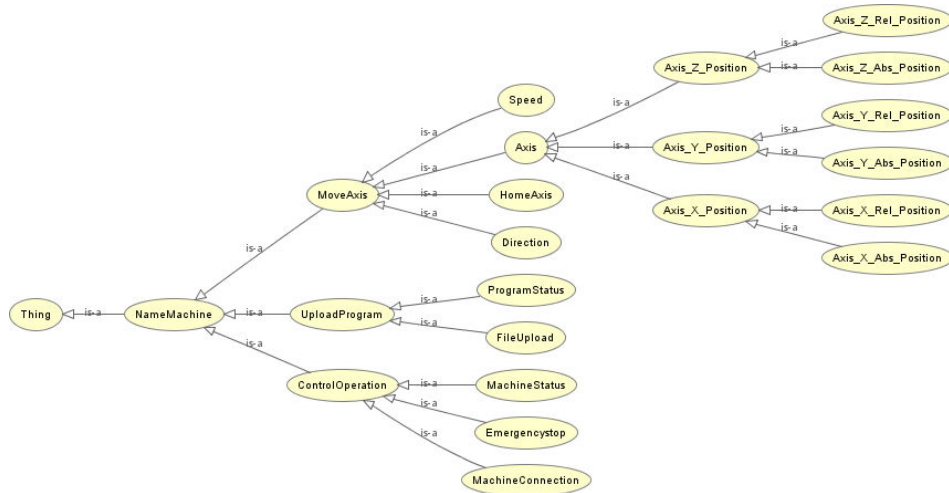


Figure 6.23 - Co-created Ontology between B1⇔B2

Concepts		
New	Mapped	Removed
	Operation ⇔ Operation_Controls SpeedDirection ⇔ Speed_orientation ⇔ Set_Speed_and_Direction	Size, Data, Conection, objects ImageBox, Label, TextEdit, Combobox, Buton, LAN, Wireless
Total of terms: 15		

Table 6.17 - C1⇔C2 Co-Creation

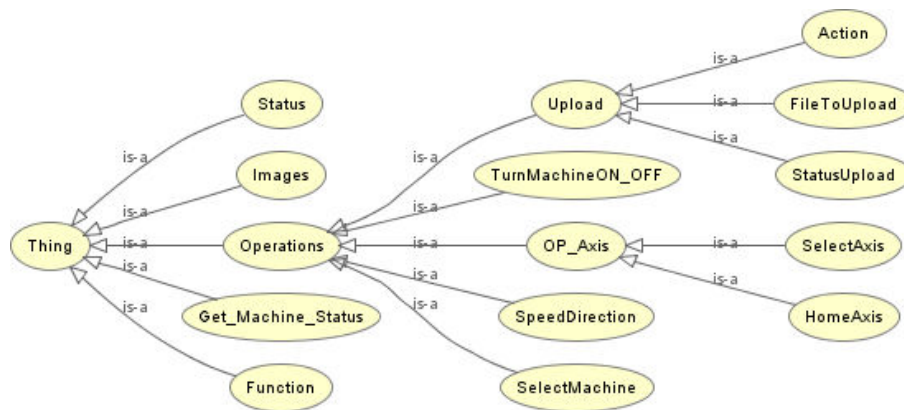


Figure 6.24 - Co-created Ontology between C1⇔C2

Concepts		
New	Mapped	Removed
	Home ⇔ Home_X	Absolute
	⇔ Home_Y	Relative
	⇔ Home_Z	Axis_position
	Move ⇔ Move_X	Speed
	⇔ Move_Y	done
	⇔ Move_Z	waiting
	Machine ⇔ Machine_On	X,Y,Z
	⇔ Machine_Off	work
Total of terms: 16		

Table 6.18 - D1 ⇔ D2 Co-Creation

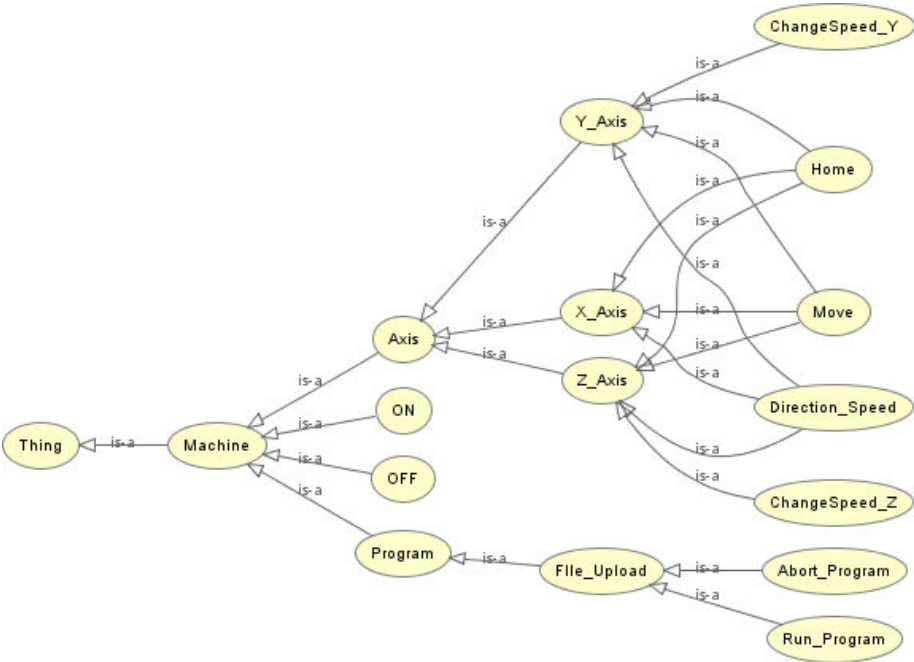


Figure 6.25 - Co-created Ontology between D1 ⇔ D2

Concepts		
<i>New</i>	<i>Mapped</i>	<i>Removed</i>
	Operation_Controls ⇔ ControlOperation Home_Axis ⇔ HomeAxis	Abort, RunProgram Waiting, Online, Offline On, Off, X,Y,Z Select_Axis, Axis Select_Program_to_Load Select_Machine, OP_Axis Turn_Machine_ON_OFF Set_Speed_and_Direction Get_Machine_Status Data, Size, Status Connection, Images Function, Lan, Wireless
Total of terms: 19		

Table 6.19 - B2⇔C1 Co-Creation

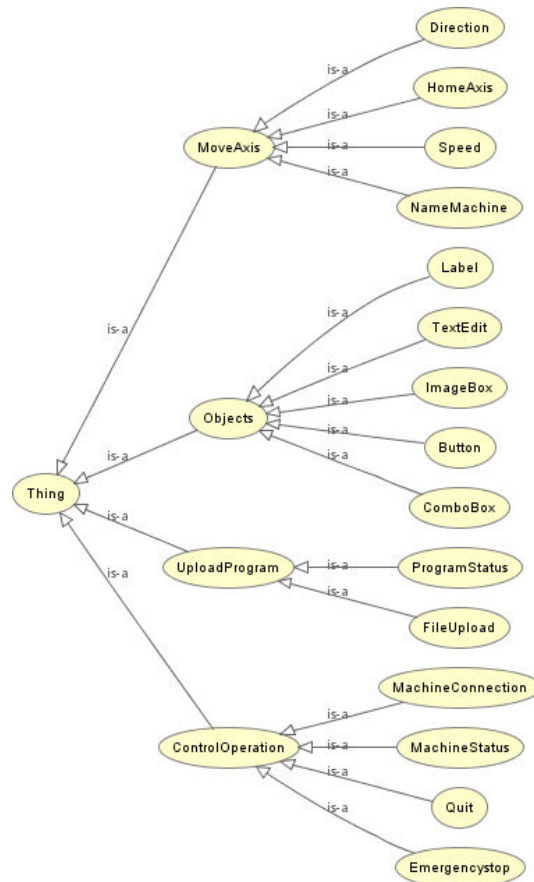


Figure 6.26 - Co-created Ontology between B2⇔C1

Concepts		
New	Mapped	Removed
	ProgramStatus ⇔ Program_Status ⇔ Estado Estado_Programa ⇔ ProgramStatus	Conexao, Peca, Peca_elettronica Peca_Mecanica
Total of terms: 21		

Table 6.20 - B1⇔C2 Co-Creation



Figure 6.27 - Co-created Ontology between B1⇔C2

Concepts		
New	Mapped	Removed
	Operation_Controls ⇔ ControlOperation Home_Axis ⇔ HomeAxis Estado ⇔ Maquina Controlos ⇔ Controlos_de_Operações Funcao_Programa ⇔ Seleccionado_Programa Carregar_Programa ⇔ Ficheiro_de_Upload Velocidade ⇔ Velocidade_de_Direcção Botao_de_Emergencia ⇔ Emergencia Conexão_da_Máquina ⇔ Ligar/Desligar On/Off ⇔ ON On/Off ⇔ OFF Online/OffLine ⇔ Online ⇔ OffLine Estado_de_Programa ⇔ Estado_Programa	Controlos, Direcção, Movimento, Posicao, Programa
Total of terms: 29		

Table 6.21 - A1⇔A2 Co-Creation

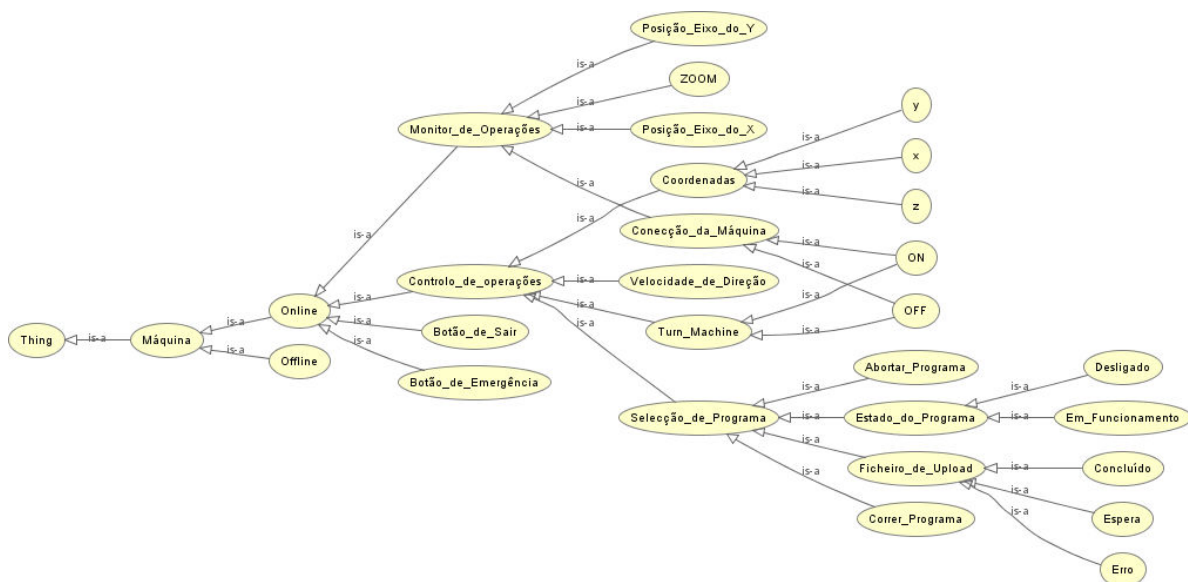


Figure 6.28 - Co-created Ontology between A1⇔A2

In this case the co-creation used the mapping as an essential tool and almost resulted in the acceptance of initial ontology of A2, as shown in Table 6.5 (pag. 152).

6.6.10 Findings

If we analyze each new ontology, easily we found that, as a complement to the initial ontologies, where the object *User Interface (UI)* was described, and the achieved mapping as a result of the analysis and experience of each user - *User Experience (UX)*, some terms were removed, others were mapped and new terms emerged. The possibility for participants to work directly in the analysis and discussion of their interpretations, allowed redirecting their different "points-of-view" – *User Pragmatics (UP)* or *Information Field (IF)*. This was made possible by communicational channels that were established between them (the conversation allowed the jointly analysis in person of created ontologies).

The following tables try to represent the mapping of terms achieved from early stages of the experimentation, i.e., the various terms that were considered equivalent (the grey area), as well as the terms that resulted in new maps, new terms and even removed or ignored (the brown area).

Just for help their interpretation, different background colors were used. New terms are represented with green color, new mappings with yellow color and removed elements with strikethrough font, as shown in next legend:

New	New terms
Mapp	New mapps
Removed	Removed elements

Let us look at the case of the co-creation between A1 and A2 participants (Table 6.22):

	UX								UP				
	Máquina	Controlos_de_Operações	Selecao_do_Programa	Ficheiro_de_Upload	Velocidade_de_Direcção	Botao de Emergencia	Conexão_da_Máquina	ON	OFF	Online	Offline	Estado_do_Programa	Programa
Maquina_Ubiqua_Remote_Desktop	✓												
Estado	✓												
Controlos		✓											
Funcao_Programa			✓										
Carregar_Programa				✓									
Velocidade					✓								
Emergencia						✓							
Ligar/Desligar							✓						
On/Off								✓	✓				
Online/Offline										✓	✓		
Estado_Programa												✓	
Movimento													
Direcção													
Posição													

Table 6.22 - A1↔A2 Co-Creation Analysis

In this co-creation, the terms *Programa*, *Movimento*, *Direcção* and *Posição* were removed. The *Online/Offline*, *Estado_Programa* terms were re-mapped. There were no new terms.

Participants A2 and A3 were unable to co-create a new ontology. The joint analysis not led to consensus and the work from Step 4 did not result in a common and agreed conclusion.

The same happened to the pair of participants A1 and A3.

	UX								UP										
	UploadProgram	Program Status	Direction	NameMachine	MachineConnection	MachineStatus	X	Y	Z	MoveAxis	Axis_X_Position	Axis_X_Abs_Position	Axis_X_Rel_Position	Axis_Y_Position	Axis_Y_Rel_Position	Axis_Y_Abs_Position	Axis_Z_Position	Axis_Z_Rel_Position	Axis_Z_Abs_Position
Programa	✓																		
Estado_Programa		✓																	
Direccao			✓																
Posicao_relativa_X							✓												
Posicao_relativa_Y								✓	✓										
Posicao_relativa_Z																			
Posicao_absoluta_X							✓												
Posicao_absoluta_Y								✓											
Posicao_absoluta_Z									✓										
Num_Maquina				✓															
Conexao					✓														
Estado_da_Maquina						✓													
Eixo										✓									
Posicao_Eixo_X											✓								
Posicao_Eixo_Y													✓						
Posicao_Eixo_Z																	✓		
Quit																			
Offline																			
Online																			
On																			
Off																			
Abort																			
RunProgram																			
Waiting																			

Table 6.23 - B1 ↔ B2 Co-Creation Analysis

Let us look at the case of co-creation between B1 and B2 participants (Table 6.23). If we recover the results of the first stages of experimentation, this was one of the pairs where most disagreements (18) and agreements (11) have occurred. However, it resulted in 21 mapped terms in those phases. After the work together and the expected co-creation, results an ontology of 22 terms, with 10 new terms, several re-mappings and several elements removed.

	UX						UP										
	Operation_Controls	Set_Speed_and_Direction	Select_Machine	Turn_Machine_On_Off	Home_Axis	Op_Axis	Select_Axis	Size	ImageBox	Label	TextEdit	ComboBox	onButt	LAN	Wireless	Select_Program_to_Upload	
CellOperationControls	✓																
Speed_Orientation		✓															
Direction_Status		✓															
SelectMachine			✓														
Turn_MachineON/OFF				✓													
Status				✓													
HomeAxis					✓												
Orientation_Axis						✓											
Axis						✓											
Direction_Status				✓		✓											
Operations	✓																
SpeedDirection		✓															
SelectAxis							✓										
Speed																	
MoveAxis																	

Table 6.24 - C1 ↔ C2 Co-Creation Analysis

Let us take a look at the results of the co-creation of the pair C1 and C2 (Table 6.24). In the early stages of the work appeared to be the pair with lowest numbered of disagreements (7) and agreements (5). They produced a total of 22 mapped terms. After co-creation, results one of the simpler ontologies, with 15 terms only, where 10 terms were once again recognized as equivalent (and so they were mapped), and some elements removed.

Considering now the pair D1/D2, at the end of the first three steps, only 8 disagreements and 6 agreements occurred. After their co-creation, from the total of 17 mapped terms, results a new ontology with only 16 terms. There were 3 new terms (*Home*, *Moves* and *Direction_Speed*) that resulted in new mappings, and 14 terms were removed (Table 6.25).

	UX						UP														
	Axis	Turn_Machine	Speed	File_Upload	X	Y	Z	ON	OFF	Direction_Speed	Home	Move	Done	Waiting	Absolute	Relative	Major_quezero	Minor_quezero	Axis_position	Select_machine	Operation_controls
Axis	✓																				
Machine On		✓						✓													
Machine Off		✓							✓												
Home_X					✓						✓										
Home_Y						✓					✓										
Home_Z							✓				✓										
Change_Speed_X			✓																		
Change_Speed_Y			✓																		
Change_Speed_Z			✓																		
Program				✓																	
Move_X												✓									
Move_Y												✓									
Move_Z												✓									
Emmergency_ON																					
Emmergency_OFF																					
X_Work																					
Y_Wprk																					
Z_Work																					

Table 6.25 - D1↔D2 Co-Creation Analysis

Looking now at the co-criation of B1/B2 pair (Table 6.26); from the analysis of the table we can conclude that:

- In individual interpreting work, several terms were mapped, but there prevailed some disagreements.
- In the jointly interpretation resulted more mapped elements (*Program_Status*, *ProgramStatus*), as well as the elimination of some of them (*Conexao*, *Peca*, *Peca electronica*, *Peca Mecanica*), getting reduced to insignificant the disagreements.
- No new terms arose.

	UX									UP			
	<i>Estado</i>	<i>Estado-da-Conexao</i>	<i>Estado-da-Maquina</i>	<i>Estado_do_Programa</i>	<i>Velocidade</i>	<i>Direcao</i>	<i>Eixo</i>	<i>Programa</i>	<i>EstadoPrograma</i>	<i>Conexao</i>	<i>Peca</i>	<i>Peca_electronica</i>	<i>Peca_Mecanica</i>
<i>Status</i>	✓												
<i>OnLine</i>		✓	✓										
<i>OffLine</i>		✓	✓										
<i>EndProgram</i>				✓									
<i>Quit</i>				✓									
<i>Stop</i>				✓									
<i>Speed</i>					✓								
<i>SpeedOrientation</i>						✓							
<i>OrientationAxis</i>							✓						
<i>Orientation</i>						✓							
<i>MoveAxis</i>							✓						
<i>Upload</i>								✓					
<i>StatusUpload</i>									✓				
<i>Program_Status</i>	✓												
<i>ProgramStatus</i>	✓			✓									

Table 6.26 - B1 ↔ C2 Co-Creation Analysis

Let us see what happened with the co-creation of the pair B2/C1. It was the only pair where there were any agreement (12 disagreements) in the early stages of the work and the one that took more time at every working step. From the 49 terms that comprised the two initial ontologies, only 11 were mapped at the end of the first 3 steps. The jointly work provided the co-creation of an ontology with 20 elements, having 4 new mappings and 26 elements removed. It was also the group where were removed more elements of the initial ontologies.

Also note that there has been no new term or mappings; just removed terms. Table 6.27 shows these results.

	UX												UP							
	ControlOperation	MoveAxis	HomeAxis	NameMachine	UploadProgram	MachineStatus	Offline	Online	Direction	Speed	X	Y	Z	RunProgram	Abort	Waiting	Online	Offline	On	Off
Operation_Controls	✓																			
OP_Axis		✓																		
Home_Axis			✓																	
Select_Machine				✓																
Select_Program_to_Upload					✓															
Get_Machine_Status						✓														
Turn_Machine_ON_OFF							✓	✓												
Select_Axis											✓	✓	✓							
Set_Speed_and_Direction									✓	✓										
Wireless																				
Size																				
Axis																				
Data																				
Status																				
Function																				
Connection																				
LAN																				

Table 6.27 - B2↔C1 Co-Creation Analysis

Let us now analyze the ontology resulting from collaboration between the three participants: A1, A2 and A3. These participants just crossed in the last stage of experimentation. There has not been direct mapping and interpretation among the three elements, before co-creating. However, the resulting work in this step deserves a common treatment to the other groups.

Table 6.28 shows the set of terms mapped in all three ontologies. It does not set out the terms that have not been mapped. By analyzing the different tables it is easy to verify that some terms (orange color) of the participant's taxonomy A1 map with terms of A2 and A3; similarly, terms of A2 (blue color) map also with A1 and A3; finally, terms of A3 (green color) map with both A1 and A2;

After worked together (steps 4 and 5), several other terms were understood as "equivalent" and therefore mapped (yellow color in Table 6.29) and new terms arose (green color in Table 6.29). Several terms have also been ignored or removed in final taxonomy.

		A2							
		Máquina	Controlos_de_Operações	Selecao_do_Programa	Ficheiro_de_Upload	Velocidade_de_Direcção	Botao_de_Emergencia	Conexão_da_Máquina	
A1	Maquina_Ubiqua_Remote_Desktop	✓							
	Estado	✓							
	Controlos		✓						
	Funcao_Programa			✓					
	Carregar_Programa				✓				
	Velocidade					✓			
	Emergencia						✓		
	Ligar/Desligar							✓	
	ON/OFF							✓	✓

A2								A3	
Controlos_de_Operações	Ficheiro_de_Upload	Posicao_eixo_X	Posicao_eixo_Y	Coordenadas	Velocidade_de_Direcção	Estado_da_Máquina	Conexão_da_Máquina		
✓									
	✓								
		✓							
			✓						
				✓					
					✓	✓			
								operation_controls	
								file_upload	
								axis_position	
								Axis	
								Speed	
								Machine_Conexion	
								turn_machine	

A1						A3	
Controlos	Posicao	Seleção_de_Maquina	Velocidade	Carregar_Program	Ligar/desligar		
✓						operation_controls	
	✓					axis_position	
		✓				turn_machine	
			✓			speed	
				✓		file_upload	

Table 6.28 - Mapping between A1 ⇔ A2 ⇔ A3

	Máquina	Controlos_de_Operações	Seleccao_do_Programa	Ficheiro_de_Upload	Velocidade_de_Direcção	Botao_de_Emergencia	Conexão_da_Máquina	ON	OFF	Posicao_eixo_X	Posicao_eixo_Y	Coordenadas	Monitor_de_operações	Botão_de_Sair	Turn_Machine	Zoom	Estado_do_Programa	Abortar_Programa	Correr_Programa	Erro	Espera	Concluido	Maior_que_0_menor_1000	Menor_que_0_e_maior_que_1000
A1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓												✓	✓
A2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
A3		✓		✓	✓		✓	✓	✓	✓	✓	✓											✓	✓
		New term						New Mapp																

Table 6.29 - A1 ⇔ A2 ⇔ A3 Co-Creation Analysis

Table 6.30 summarizes the most relevant details resulting from the different stages of experimentation.

From the initial interpretation of the study object (*User Interface*) resulted an ontology with a different taxonomy (in numbers and hierarchy of terms) among all participants. From the individual interpretation (steps 2 and 3), and given the experience of each one (*User Experience*), resulted differing opinions. With jointly work (steps 4 and 5) was possible to analyze the prospect of each (*Information Field*) and each one might justify or argue, at that moment and personally (*User Pragmatics*), his interpretations. There was thus obtained consensus and decisions that sustain the final ontologies that, in addition to being substantially different from the initial, are aligned with the joint perspective.

	UI	UX				UP				
	STEP 1	STEP 2		STEP 3		STEP 4		STEP 5		
	Taxonomy	Desagee	Agree	Desagee	Agree	Desagee	Agree	New	Mapped	Removed
B1-B2	22	18	11	3	5	0	7	0	11	8
		Mapping						Taxonomy		
		21						22		
B2-C1	25	12	0	5	4	3	6	4	26	
		Mapping						Taxonomy		
		11						20		
B1-C2	22	6	14			4	4	5	4	
		Mapping						Taxonomy		
		16						21		
C1-C2	21	7	5			0	4	5	10	
		Mapping						Taxonomy		
		22						15		
D1-D2	23	8	6			0	8	11	12	
		Mapping						Taxonomy		
		17						16		
A1-A3	20	11	4			0	9	NCO ^(*)		
		Mapping								
		11								
A2-A3	36	11	0			0	9	NCO ^(*)		
		Mapping								
		14								
A1-A2	36	7	6			0	11	13	5	
		Mapping						Taxonomy		
		18						29		
A1-A2-A3								2	11	
								Taxonomy		
								27		

Table 6.30 - Experiment Results Summary

^(*)NCO - Not Created Ontology

Following more or less formal processes (ontologies) and using the appropriate platforms (Protégé SW), was possible to get a set of concepts that were not sufficient to clearly describe the subject. Many discrepancies existed.

The phase of the experimentation with Pragmatic support (steps 4 and 5) contributed clearly to the explanation of the disagreements detected among participants. At the end it was possible to define new ontologies that describe the initial interface in which both authors are in agreement.

6.7 Relevance to the Communicational Architecture

6.7.1 In Transactional architectures

The way how the inconsistencies (agreements and disagreements) were debated ensured that proceedings could flow and reach "a successful conclusion". From a (more) personal initial definition reached a (more) consensual definition at the end, but not only with automatic methods of analysis and processing of concepts. It was necessary that the intervenientes could intervene in a way to clarify eventual doubts or questions about decisions taken so far. This was achieved through communication channels, in this case the face-to-face between the people involved.

The need for such a process to be supported by an application that aims to be ubiquitous and based on a communicational architecture should likewise permit, on the one hand, the presentation of appropriate terms, and by other, disposal mechanisms to manage conflicts that may arise.

In a context where you want to be multiple-domain and support multiple users, the probability of conflict is high, in particular when dealing with different specialty areas.

The use of taxonomies or ontologies will be essential tools to manage potential inconsistencies. The need to exploit existing ontologies (Kontchakov, Wolter, & Zakharyashev, 2010) depends on however their interoperability (Fonseca, Câmara, & Monteiro, 2006; Guarino, 1998), i.e., depends essentially on the ability to map terms (Kalfoglou & Schorlemmer, 2003) between taxonomies of the different ontologies. On the other hand the ontologies should be able to "learn" and include new terms that were related or mapped (Maedche & Staab, 2001).

In a *cloud-based* architecture the SOA patterns contribute significantly to minimize the gap between business processes and technologies, and promote the reuse of existing (legacy) applications through services that represent them. But if on the one hand we can accept that this application integration and interoperability of processes as reasonably achieved, it still remains

the ability to know properly what they really perform and the quality with which they do so. The essential idea of SOA patterns is based on the ability and easiness to disseminated, found, integrate (or combine) and executed those services.

Doing a quick retrospective, as happened with the web (merely syntactic) where all the information (pages) is "related" only by hyper references between it, and where existing information is dispersed and sometimes inconsistent, requiring robust search mechanisms supported only by the machine, with results far below expectations, there was a need to relate that information by the meaning (or sense) it could represent (or have) in order to allow better utilization by their users (humans). This relationship of meanings resulted from the analysis of terms that exist in the information and was mapped in meta-information. For example, if in a web page is the word "doctor", his content must be related to health or medicine. Thus arises the Web Semantics.

The same phenomenon occurred with web services. They proliferate on the web and it was necessary to manage them. The *SWSI - Semantic Web Services Initiative*, bringing to web services the principles of semantic web to web, trying to maximize the autonomy and dynamism in the process that involves the use of web services: the publication, discovery, negotiation, composition, use, monitoring and other (McIlraith, Son, & Zeng, 2001). This scenario increases exponentially with the SOA and now with cloud based services.

But all these processes are no more than attempts to improve the search of terms, either on the used information (Figure 6.29 (a)) and on the technologies that are used in the implementation (for example, the use of agents on DAML⁴⁴ Figure 6.29 (b)).

6.7.2 For Communicational architectures

The search of terms (*Information Brokering*) represents a set of processes that aims to collect a set of grammatically "related" terms: synonyms, hyponyms, homonyms, etc. The use of ontologies, taxonomies, thesaurus, topic maps, etc., is common useful tools in these processes. But the experimentation here carried out demonstrates however that the relationship (i.e., interoperability) between terms is not always easy, and map terms cannot be possible only with the use of these tools. It was demonstrated that this can be possible with the use of other

⁴⁴ DAML – DARPA Agent Markup Language

complementary tools, especially those that allow "brokers" (participants, in this case) to examine and interact according to their information field. Direct conversation between participants resulted in co-creation, i.e., a jointly creation, of several new terms and the final creation of consensus ontologies (that had been initially interpreted as non-consensual).

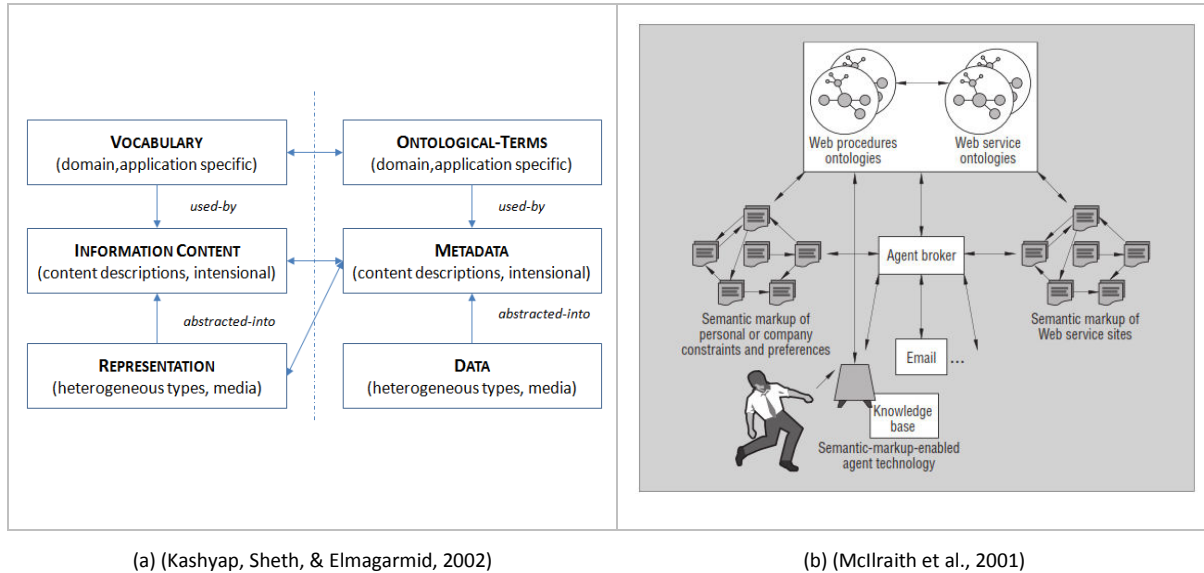


Figure 6.29 - Search of terms: a) Vocabulary Broker; c) Agents in Semantic Web Services

In the context of a Market of Resources, the broker (*Resources Broker*) has as its main task "to find" resources. But to ensure that resources are found, they must be properly registered in the Market of Resources. The resources "enter" on the market with meta-information for terms of specific domain ontology which they belong. These cataloging processes (or classification) shall allow: a) reuse existing domain ontology terms; b) reuse existing terms in different domain ontologies (ontology library) and c) to create new terms.

Following this process of cataloguing, new relationships are established between terms that represent in the degree of interoperability between them. It is here that we believe in Pragmatics as a way to achieve effective processes.

Many of the difficulties in the process of mappings (set relationships) based only on computational processes can be resolved through direct communication channels between providers and clients, leading even to the possibility to achieve the co-creation of new terms. Such process allows defined ontologies to constantly evolve, in terms and relationships between them.

In a functional perspective, it is necessary to navigate in a global network of terminology (Regina, Marta, & Claudia, 2001) associated to functional units (resources) that they relate to. In practice, implement a brokering service in ontologies - *Ontology Brokering Service* (Figure 6.30).

It is necessary also to have mechanisms which intervene and assist in mapping or defining interoperability between terms, using communication channels to ensure Pragmatics. This is the *Pragmatics Mapping Service*. In practice, it assists in the classification of resources so that, when necessary, can be established forms of direct communication between stakeholders.

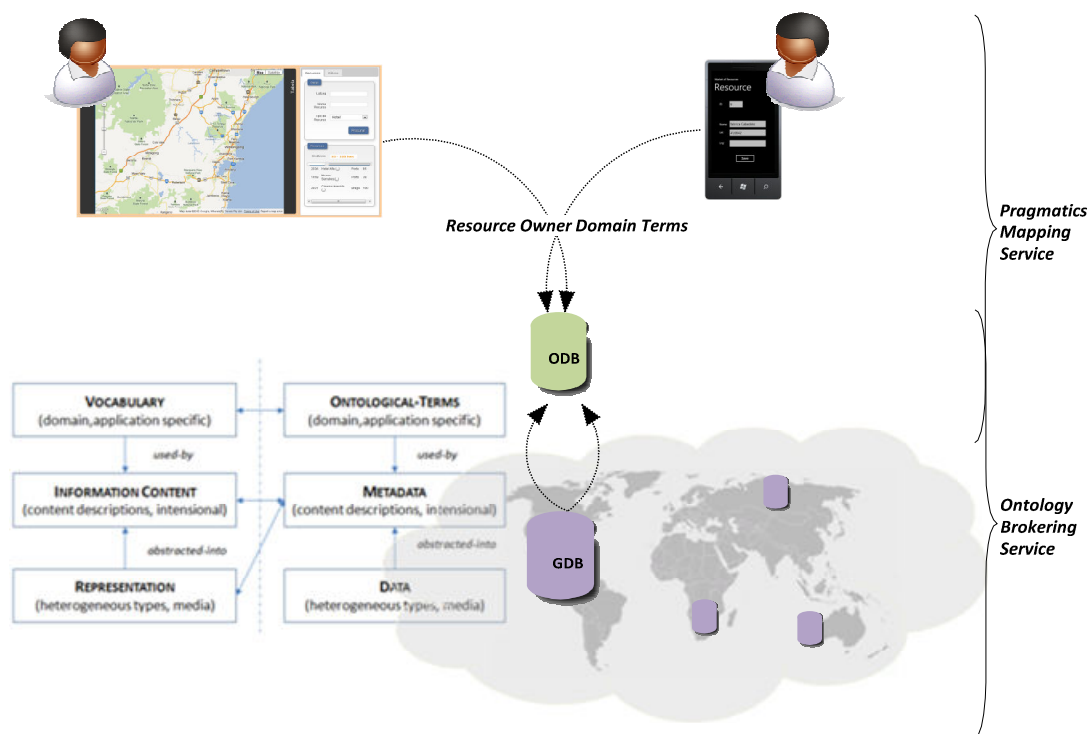


Figure 6.30 - Pragmatics and Ontology Brokering Service

For the *Ontology Brokering Service*, the information Brokering model designed by (Kashyap & Sheth, 1997) represents an interesting starting point but there is a need to expand in services and on the platform that supports it. For example, make it able to handle JSON format (in addition to XML), format for brokering on data and meta-data in full expansion.

The success of the entire process will be dependent on the existence of a wizard that should help the resource classification (meta-information definition) during resource registration (Figure 6.31 (I)). The terminology used by resource providers (*Resource Owner Domain Terms*) is properly

processed and catalogued in databases of terms (*ODB - Ontologies Database*) (Figure 6.31 (V)). After catalogued, the resource is registered in the market (Figure 6.31 (II)).

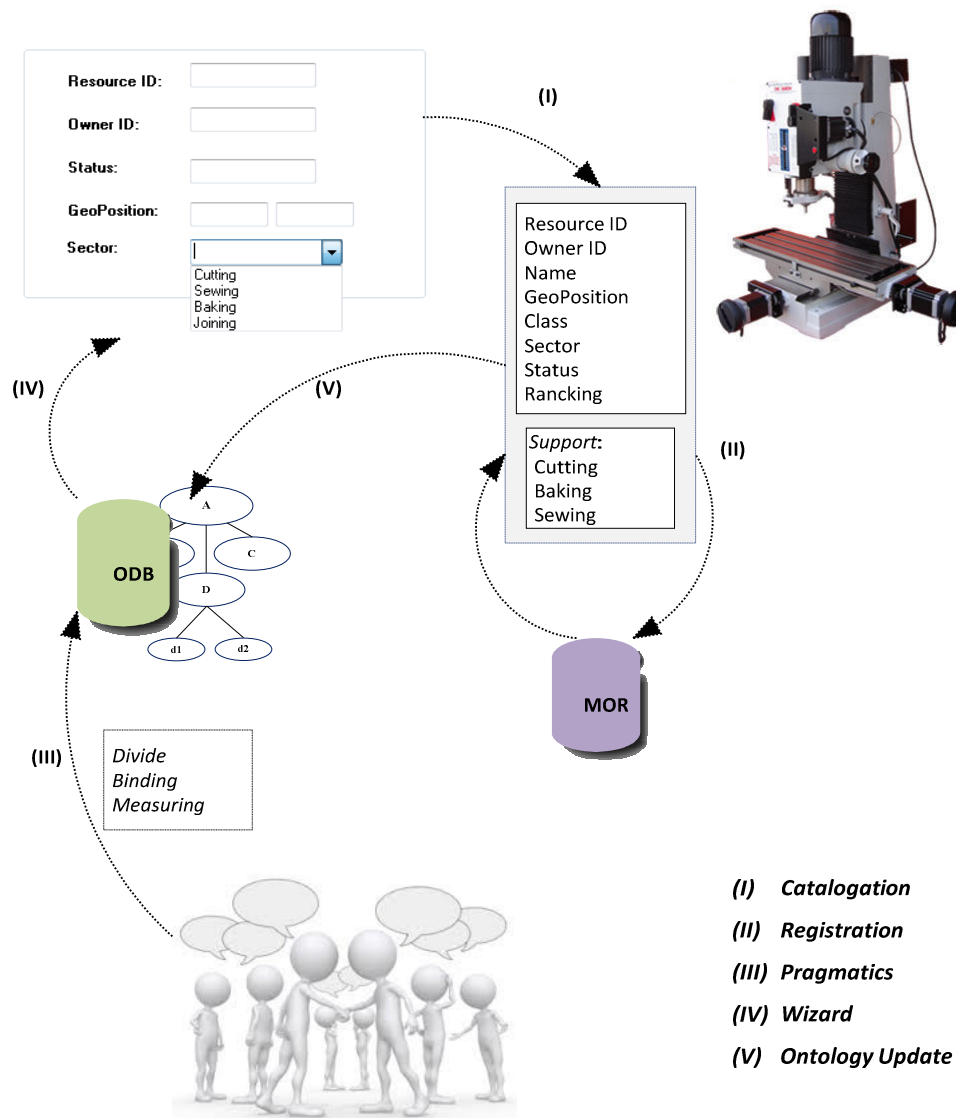


Figure 6.31 - Applied Communicational Architecture

The *Ontology Brokering Service* is a dynamic and expansive process based on cloud-based services. The information coming from multiple existing information sources (*GDB - General Databases*) is catalogued with sufficient meta-information to ensure an efficient support of any query operations. Search engines (and brokering) work first on the meta-information (*MetaData Broker*). Next, if necessary, forwards them to the associated terminology, equivalent or related, existing in the ODB taxonomies.

The applications wizards use terminology that exists in the ODB (Figure 6.31 (IV)). Whenever necessary, the direct user participation helps to solve any "conflicts" (Figure 6.31 (III)).

The architecture of Figure 6.31 outlines the entire process.

6.8 Conclusions

The common context put the starting point as a semantic interoperability or integration problem. However it is thus established that semantics by itself cannot ensure the correct relation between concepts. Multiple and distinct semantics are not liable to be easily integrated. To effectively integrate semantics they are necessary mechanisms that allow direct participation of all stakeholders. Co-creation processes are essential. Communicational channels are the bases.

7 IMPLEMENTATION OF PROTOTYPES FOR THE PROPOSED ARCHITECTURE

The proposed architecture (modeled in Chapter 5) fits those applications that are able to support activities with the following characteristics:

- Dynamics processes
- High number of participants
- Multiple supporting platforms
- Multiple devices
- Fiability
- Dynamic Reconfiguration

The architecture technological base, among others requirements, is based on the need: a) to support autonomous and heterogeneous entities (processes, applications, users, others), that need to collaborate (hence the need of appropriated middleware); b) to assure services ubiquity; c) to assure the portability between platforms and devices; d) to assure pragmatics to the user, i.e., mechanisms or tools that encourage co-creation processes between users; and e) to assure responsiveness and multimodal interfaces.

Following we present two prototype implementations that show the applicability of proposed architecture to different economic areas: *Cloud Ubiquitous Manufacturing* and *Dynamic Tourism*.

7.1 Cloud Ubiquitous Manufacturing

We could saw in Chapter 2 that Ubiquitous Manufacturing business model brought the quick reaction to market changes, and the high availability and capacity to effectively support changed requirements, as the main sustainability criterion.

Therefore, we could saw that their management systems - *Ubiquitous Manufacturing Systems (UMS)* – could be (and must be) a set of well integrated distinct au autonomous components.

Thus it can be possible if the efficient interoperability between participating resources (people, machines, time, services, etc.) is ensured.

7.1.1 Entities

Manufacturing is a complex and very dynamic economic activity, sensible to continuous changes due to multiple internal (resources failure, insufficient resources, production errors, etc.) or external (climatic conditions, legal regulations, others) application factors.

Manufacturing process involves a production plan, a set of resources and needed raw material, essentially. Many causes can imply the production process reconfiguration, mainly those related with resources availability and capacity. So, we identify three main entities in these processes:

- The Client (which wants the product)
- The Enterprise (that provide a set of resources)
- The Market of Resources (that mediate the company/client relation, ensure the supply of resources).

Having the architecture applicability as the main goal, and considering the complexity of the area and multiplicity of contexts,

Given the complexity of the area and multiplicity of contexts, and considering the architecture applicability as the main goal, we decide to model the entities only with the information considered essential to the services we wanted to demonstrate:

- a) Resource Registration
- b) Communicational Channels Registration
- c) Tasks that Resources can execute
- d) Production Plan definition
- e) Search Resources by state, classification, sector, georeferenced information
- f) Changing the status of the resource
- g) Production Plan Reconfiguration
- h) Resources Georeferencing
- i) Production Plan Georeferencing
- j) Use of the Communicational Channels

7.1.2 Relational Model and Class Diagram

Following we present the relational model of the main entities. It was only specified the information considered essential to demonstrate the specified services.

Resources, Tasks and Communicational Channels

The Resource is the main entity of the system and has information of multiple types:

- Specification data* (name, owner, etc.);
- GeoPosition* for its map localization;
- State and Sector of Activity*;
- Communicational Channels* (*channelResource*), responsible for Pragmatics services support.

Each resource has a set of tasks that it can execute (*ResourceTask*) (Figure 7.1).

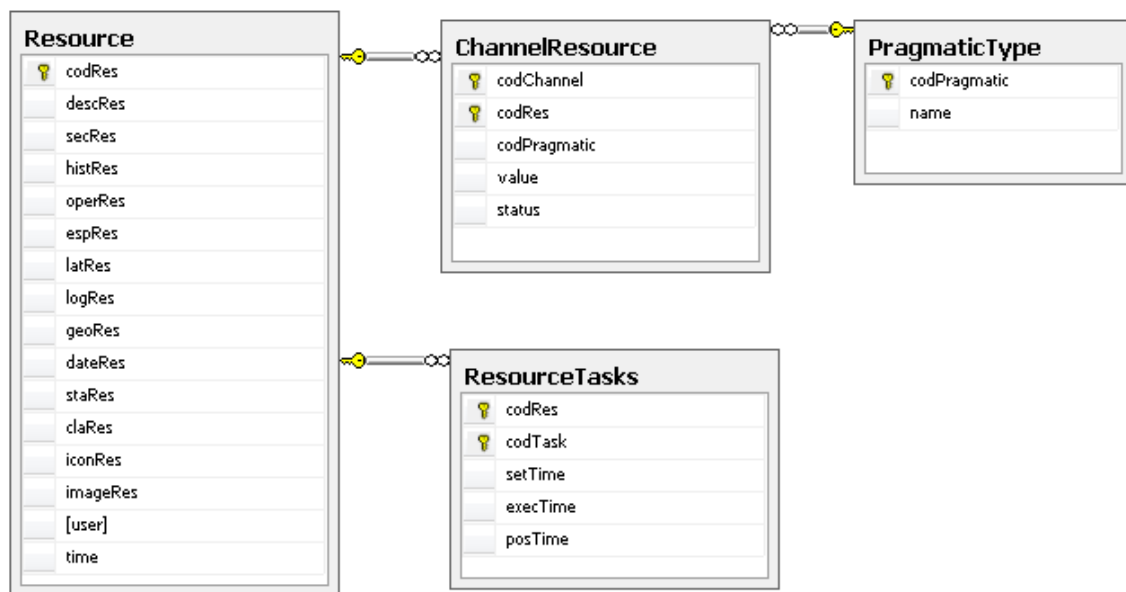


Figure 7.1 - Relational Model for Resource, Task and Communicational Channel

Production Activity

It corresponds to a set of sequential and ordered tasks (*ActivityTasks*). However, when a particular activity is associated to a particular client, it must be present a set of complementary data:

- a) *Temporality data* (when each task starts and ends);
- b) *Monitoring data* (task execution state: in course, completed, etc.) in *AffectedActivity* entity;
- c) *Related resources*, i.e., in what resources will be executed each task, in *ActivityTaskResource* entity (Figure 7.2).

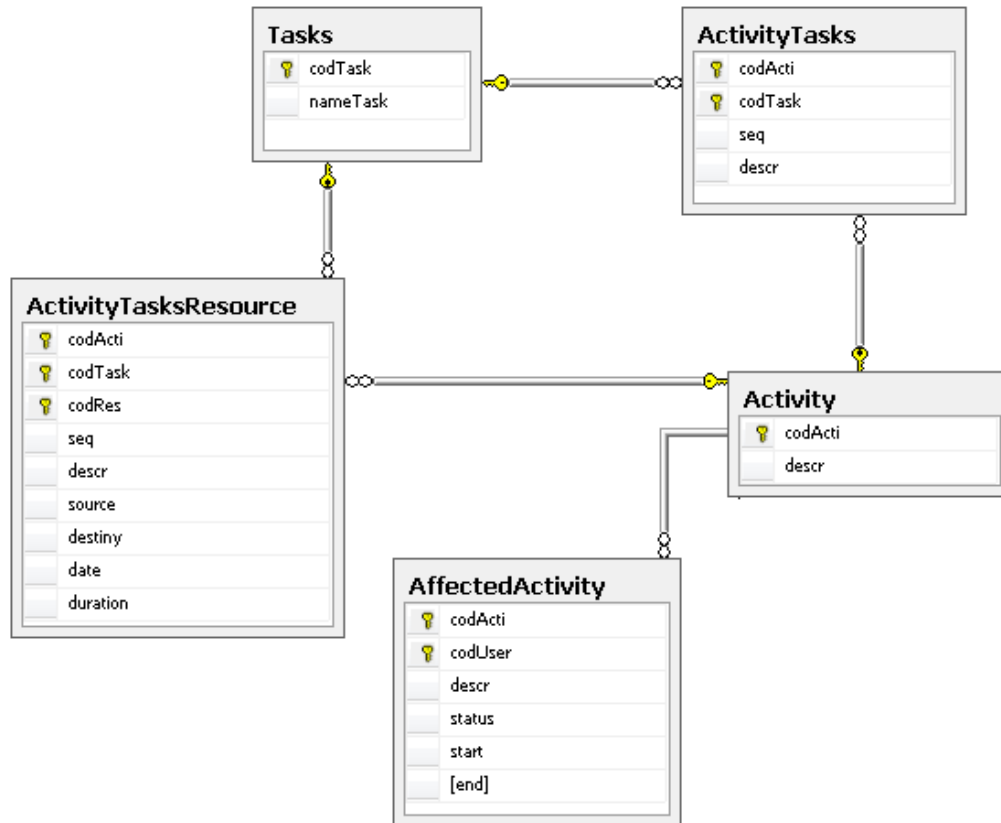


Figure 7.2 - Relational Model for Production Activity

Market of Resources

Objectively, a *Market of Resources (MR)* is a repository of resources and a set of services that operate on it (Cruz-Cunha, Putnik, Silva, & Santos, 2005). This *Manufacturing Market of Resources (MMR)* includes, on the one hand, services to manage manufacturing resource's state, and advanced searching services, the other. The latter underlie the service brokering over resources.

Essentially, the MMR has services for: a) resource selection; b) resources registration; c) XML/JSON resources serialization and d) updating resources.

As resource selection services were specified the criteria location, status, classification and business sector with which they are associated. Regarding the resource allocation it is possible to know the tasks that each resource can support and the tasks that are allocated to it. Considering monitoring and interaction it is possible to define and select the available communicational channels (*Pragmatics Channels*).

Figure 7.3 shows an excerpt of the C# Class Diagram of these main system entities.

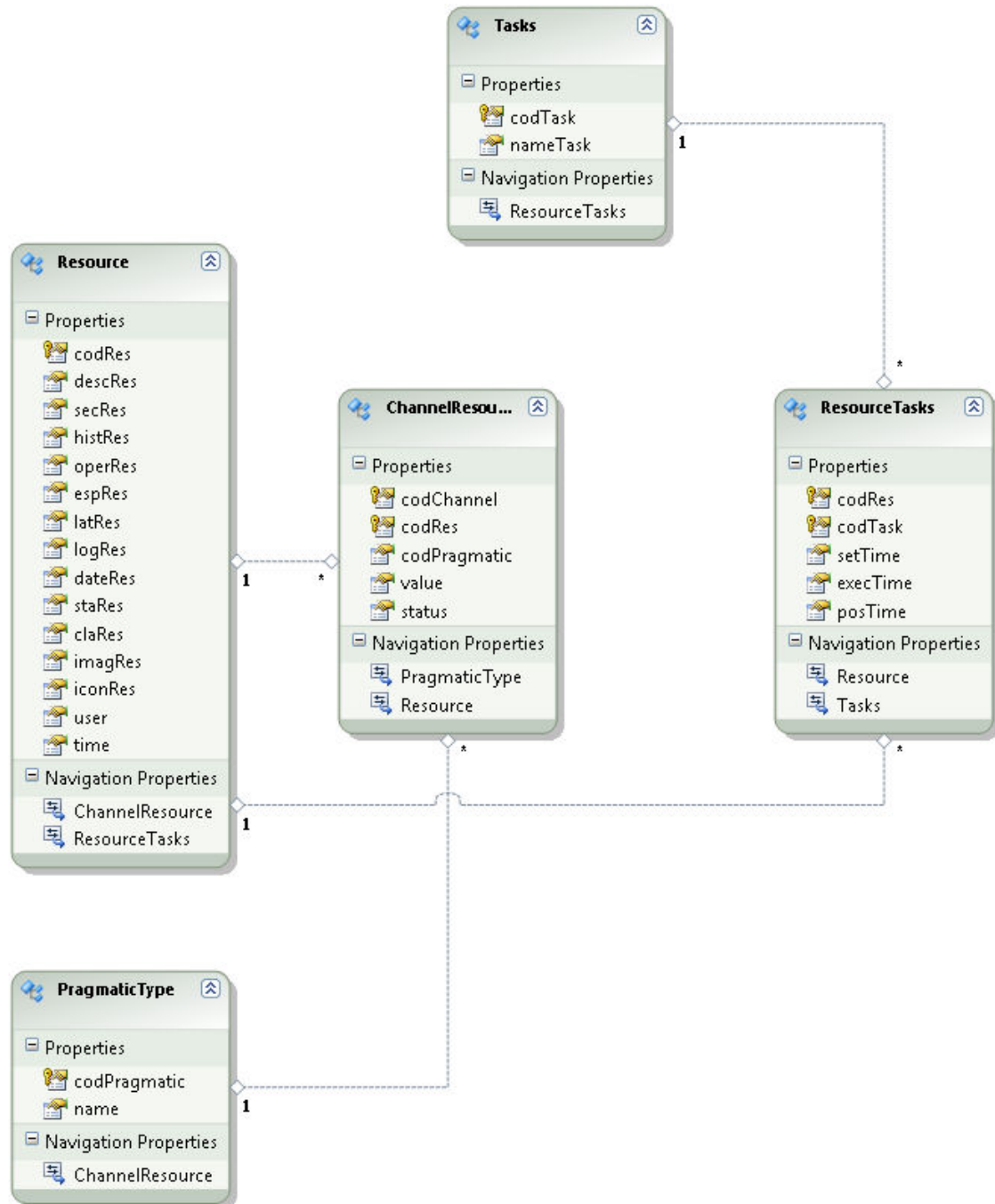


Figure 7.3 - Class Diagram: Resources, Tasks and Channels

The MMR is an entity with a fully autonomous behavior and able to be integrated in any application (multiple distinct areas) that needs to manage and use a set of resources. This model of "behavior" provides it with the necessary and essential for its sustainability.

It is a similar model to what currently exists in social networks like Facebook, Twitter and others. The presence of resources depends on the interest of its use and, consequently, their promotion. The disclosure (or registration) of resources is an independent process.

7.1.3 Used Patterns and Anti-Patterns

The API is based on WCF services (Figure 7.4). Was developed to avoid some of the more common anti-patterns (William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick, & Mowbray, 1998), such as *CRUDy Interface* and *Loosey Goosey*, and according to patterns *Message API*, *Service Contract* (Table 7.1 and Table 7.2), *Document-Literal-Wrapped SOAP binding style* (used by SOAP and WSDL), and *Contract Centralization* (Table 7.1 and Table 7.2) that are supported by WCF framework. Also *Repository Pattern* was essential to implement local repositories (Mark Endrei et al., 2004).

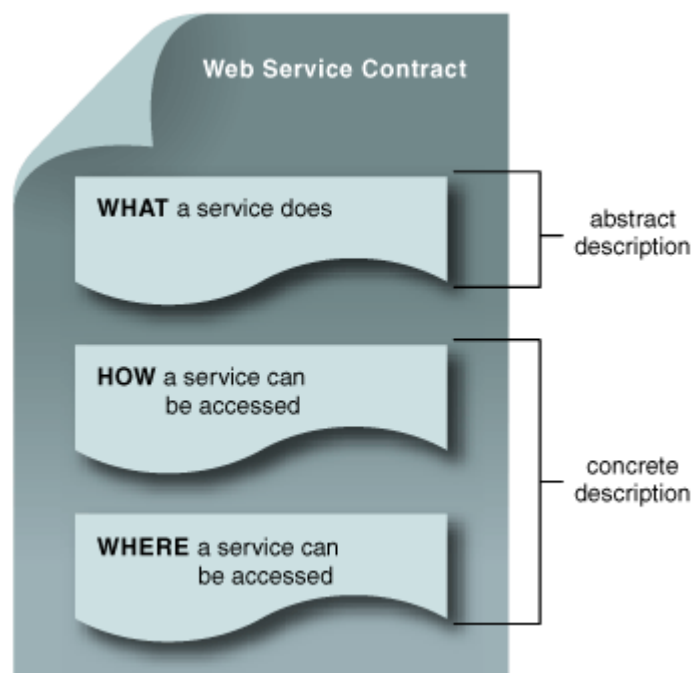


Figure 7.4 - Service Contract
(Erl et al., 2008)

<i>Service Contract</i>
<pre> [ServiceContract] [ServiceKnownType(typeof(ResourceStatusEnum))] [ServiceKnownType(typeof(ResourceClassEnum))] public interface IMarket { [OperationContract] int NewResource(Resource r); [OperationContract] int NewResourceStr(string u, string n, string lat, string lng, ResourceStatusEnum sta, ResourceClassEnum cla); [OperationContract] DataSet GetAllResources(); [OperationContract] string GetAllResourcesJson(); ... } </pre>

Table 7.1 - Service Controller/Message API Patterns implementation

<i>Data Contract</i>
<pre> /// <summary> /// Resource Managment Class /// </summary> [DataContract] [KnownType(typeof(ResourceClassEnum))] [KnownType(typeof(ResourceStatusEnum))] public class Resource { string codRes; string descRes; string setRes; Position geoRes; long user; ResourceStatusEnum staRes; ResourceClassEnum claRes; [DataMember] public string CodRes { get; set; } ... public Resource() { } } </pre>

Table 7.2 - Resource DataContract

All services are ready to be used asynchronously, respecting the *Asynchronous Response Handler Pattern* (Daigneau, 2012) and going throw the requirement of *latency minimization*.

Table 7.3 shows the client application using *GetAllChannelsResourceAsync* service asynchronously according to this pattern.

<i>Web Services Asynchronous Invocation</i>
<pre> /// <summary> /// Get Channels Resources... /// </summary> public static void GetAllChannelsResourceAsync() { MorWS.MarketClient ws = new MorWS.MarketClient(); string s = ws.GetAllChannelResource(); ws.GetAllChannelResourceCompleted += new EventHandler<MorWS.GetAllChannelResourceCompletedEventArgs>(ws_GetAllChannelResou rceCompleted); ws.GetAllChannelResourceAsync(); } /// <summary> /// Get Channels Resources Asynchronous Handler /// </summary> static void ws_GetAllChannelResourceCompleted(object sender, MorWS.GetAllChannelResourceCompletedEventArgs e) { if (e.Result != null) { ChannelsTemp = new List<Channel>(); string s = e.Result; ToListChannelTemp(s); } } </pre>

Table 7.3 - Asynchronous Response Handler Pattern

The pattern *Resource API* was essential in order to ensure that a client application can manage the data that are sent by the services. It was created a set of services that allow then to serialize data in open and nonproprietary formats. The client application can then explore this data with the desired API.

Table 7.4 shows an auxiliary method to help services to serialize to JSON their return values. Its implementation was justified to follow the mentioned *Resource API* pattern.

<i>Resource API</i>
<pre> /// <summary> /// DataSet serialization to JSON /// </summary> /// <param name="ds">DataSet</param> /// <returns>Json</returns> public static string DStoJSON(DataSet ds) { StringBuilder json = new StringBuilder(); foreach (DataRow dr in ds.Tables[0].Rows) { json.Append("{"); } } </pre>


```

        int i = 0;
        int colcount = dr.Table.Columns.Count;

        foreach (DataColumn dc in dr.Table.Columns)
        {
            json.Append("\"");
            json.Append(dc.ColumnName);
            json.Append("\":\");
            json.Append(dr[dc]);
            json.Append("\"");

            i++;
            if (i < colcount) json.Append(",");
        }
        json.Append("\}");
        json.Append(",");
    }
    return json.ToString();
}

```

Table 7.4 - Resource API Pattern: DataSet serialization to JSON

The serialized JSON data (a string) is now prepared to be explored for any client API. Table 7.5 shows a C# method exploring it to create a *Collection List*.

Exploration of Json Web Services API

```

/// <summary>
/// Convert JSON to List<Channel> using LINQ
/// </summary>
/// <param name="s"></param>
private static void ToListChannel(string s)
{
    XmlDocument doc = (XmlDocument)JsonConvert.DeserializeXmlNode(s);
    XDocument infoChannelBase = XDocument.Parse(doc.OuterXml);

    var res = from info in infoChannelBase.Descendants("Table")
    select new
    Channel(
        (int)info.Element("codRes"),
        (int)info.Element("codChannel"),
        info.Element("value").Value, (CommunicationalOptions)Enum.ToObject
        (typeof(CommunicationalOptions), (int)info.Element("codPragmatic"));
    Channels.AddRange(res.ToList<Channel>());
}

```

Table 7.5 - C# JSON Serialization

The JSON can now be easily explored and manipulated using JQuery in front-ends, for instance (Table 7.6).

JQuery API exploring Json

```
//-----
// Get Resources from Repository
//-----
function GetResources(codRecurso) {
    var marks = [];

    $.ajax({
        url: '@Url.Action("MorGetResourcesRepositoryJson", "Mor")',
        data: "codRes=" + codRecurso,
        dataType: 'json',
        cache: false,
        async: false,
        type: 'POST',
        success: function (response) {
            marks = CreateMarkersArray(response);
        },
        error: function (xhr) {
            var str = "[ERROR in GetResources; gmap.html]: STATUS: " +
xhr.status + " - STATUSTEXT: " + xhr.statusText + " - RESPONSETEXT: " +
xhr.responseText;
            alert(str);
        }
    });

    return (marks);
}
```

Table 7.6 - JQuery JSON asynchronous utilization

The API also has services able to support the *Observer Pattern in Real-Time* (Daigneau, 2012). Any change to the state of a resource is immediately known to all the other resources that are related to it. This happens, for example, whenever a communicational channel of that resource changes its status (Table 7.7).

Real-Time Resource Status

```
//-----
// Get Resources Info from ChannelRepository
//-----
public JsonResult MorGetInfoRecRepositoryJson(int codRes)
{
    List<Channel> info= new List<Channel>();
    info = ChannelRepository.Channels.FindAll(d => d.CodRes == codRes).ToList();
    return Json(info, JsonRequestBehavior.AllowGet);
}
```

Table 7.7 - Real-Time Pattern Service

7.1.4 Components

The development focused the following components of the global proposed architecture (Figure 5.21, pag. 117):

- a) Market of Resources Engine
- b) Brokering
- c) Pragmatics Engine

Two distinct applications were implemented to explore these components: a Web Portal and a Mobile Application (Windows Phone), both with multiple services:

- a) a Web Portal that:
 1. support Manager analysis;
 2. support Client resource monitorization;
 3. support Dynamic Reconfigurations
 4. allows participants direct communication
- b) a Mobile Application that allows the resource promoter to register their own resources.

In practice, all these integrated services sustain all public information of any resource (Figure 7.5).



Figure 7.5 - Public Resource Information

Let us then analyze what was explored in each of these components and how they are integrated to respond efficiently and effectively to the customer's requirements.

7.1.4.1 A – Market of Resources Engine

The *Manufacturing Market of Resources* (MMR) represents in practice, an instance of a *Market of Resources* (MR). Its functionalities are supported, in part, by the relations defined in the Database. The remaining part is supported by a set of operations that operate on it.

Since MMR will be hosted in a cloud engine, its real location will be virtual and distributed and so its handling will have to be made securely via the use of web services. Having objectively been postponed security issues and *failures* support, the implemented services are based on WCF web services and *Restful* services.

The Web Services API were created according to existing patterns, mainly Encapsulation, Service Contract, Autonomy, Latency minimization (using asynchronism), Binary message encoding of text-based data and global data (Daigneau, 2012), and format serializing (XML and JSON).

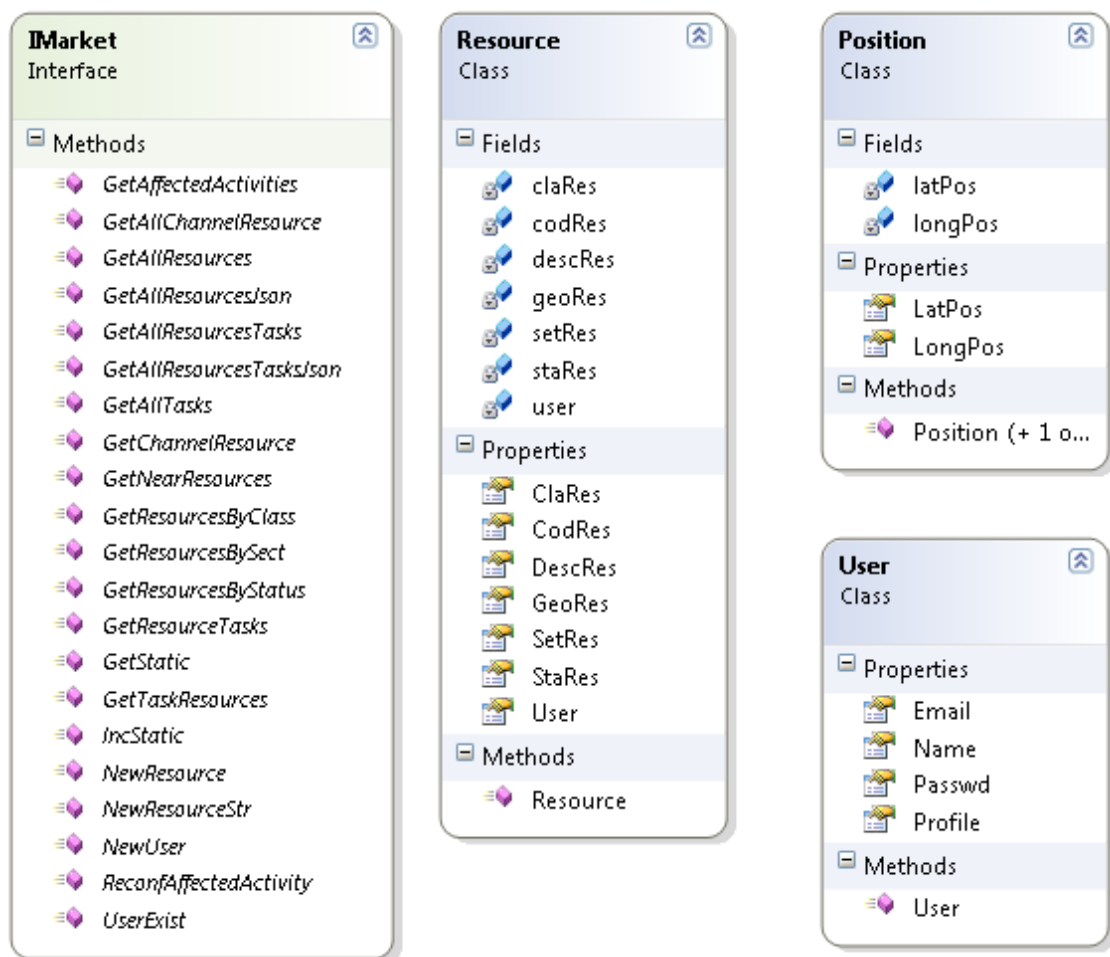


Figure 7.6 - Manufacturing Market of Resources

The Figure 7.6 shows the main classes that supports MMR. Following the standard WCF, it is possible to observe in *IMarket interface* all signatures of all implemented methods (web services).

There is also the definition of a few business objects that provide data serialization. They are *Resource*, *Position* and *User*.

As a whole it is an API able to be used by any other application. We will see later two examples that demonstrate the ease of such use.

Most services were implemented using *Transact-SQL Stored Procedures* (SP), complemented with *Triggers*, *UDF* (User Defined Functions) and the *Common Language Runtime* (CLR) Integration to ensure best performances and the use of web services from within our own SP (Pathak, 2011).

The excerpt of code of Table 7.8 shows an example of a created SP in Transact-SQL, in this case associated with the registration of a new resource.

<pre> sp_NewResource -- ===== -- MOR: Market Of Resources -- ===== -- ===== -- Author: lufer -- Create date: 05-02-2012 -- Description: Resource registration -- ===== CREATE PROCEDURE sp_NewResource @descRes nvarchar(36), @user numeric(18,0), @lat decimal(8,5), @long decimal (8,5) AS BEGIN --get last resource code Declare @tot int select @tot=count(*) from dbo.Resource -- actual date Declare @MyDate datetime set @mydate=getdate() -- new POINT from Lat/Long Declare @geo geography SET @geo= 'POINT(' + convert(varchar(100),@long)+' ' + convert(varchar(100),@lat) +')' insert into Resource (codRes, descRes, latRes, logRes, geoRes, </pre>

```

        [user],
        dateRes
    )
    values
    (
        @tot+1,          -- inc code
        @descRes,
        @lat,
        @long,
        @geo,
        @user,
        @MyDate
    )
    return @@error
END
GO

```

Table 7.8 - A Stored Procedure

This SP is used by the *NewResource* method of the API of MMR, whose excerpt of implementation is presented in the following code (Table 7.9).

```

NewResource
/// <summary>
/// Regista um novo recurso. Utiliza um Transact-SQL StoredProcedure
/// </summary>
/// <param name="r">Resource</param>
/// <returns>(int)Control Value</returns>
public int NewResource(Resource r)
{
    SqlConnection con = GetConnection();
    SqlCommand sqcmd = new SqlCommand();
    sqcmd.Connection = con;
    con.Open();

    int sta = (int)r.StaRes;    //estado do Recurso
    int cla = (int)r.ClaRes;    //classificação do Recurso

    //preparar acesso ao StoredProcedure
    sqcmd.Parameters.Clear();
    sqcmd.CommandText = "sp_NewResource";
    sqcmd.CommandType = CommandType.StoredProcedure;

    //instanciação de parâmetros
    sqcmd.Parameters.AddWithValue("@descRes", r.DescRes);
    sqcmd.Parameters.AddWithValue("@user", r.User);
    sqcmd.Parameters.AddWithValue("@lat", r.GeoRes.LatPos);
    sqcmd.Parameters.AddWithValue("@long", r.GeoRes.LongPos);
    sqcmd.Parameters.AddWithValue("@cla", cla);
    sqcmd.Parameters.AddWithValue("@sta", sta);

    //gerir transacção
    sqcmd.Transaction = con.BeginTransaction();
    try
    {
        int p = sqcmd.ExecuteNonQuery();
        sqcmd.ResetCommandTimeout();
        sqcmd.Transaction.Commit();
        return p;
    }
}

```

```
    }
    catch (Exception e)
    {
        sqcmd.Transaction.Rollback();
        throw new Exception(e.Message);
    }
    finally
    {
        con.Close();
    }
    return 0;
}
```

Table 7.9 - *NewResource* method of the API of MMR

All operations that can change the state of entities in the database are safeguarded with transactional control.

Equal treatment had all other methods that make up the API. The following schema (Figure 7.7) shows the set of available methods directly related to the management of the market of resources, not being present, however, those auxiliary methods or stored procedures developed for complementary purpose.

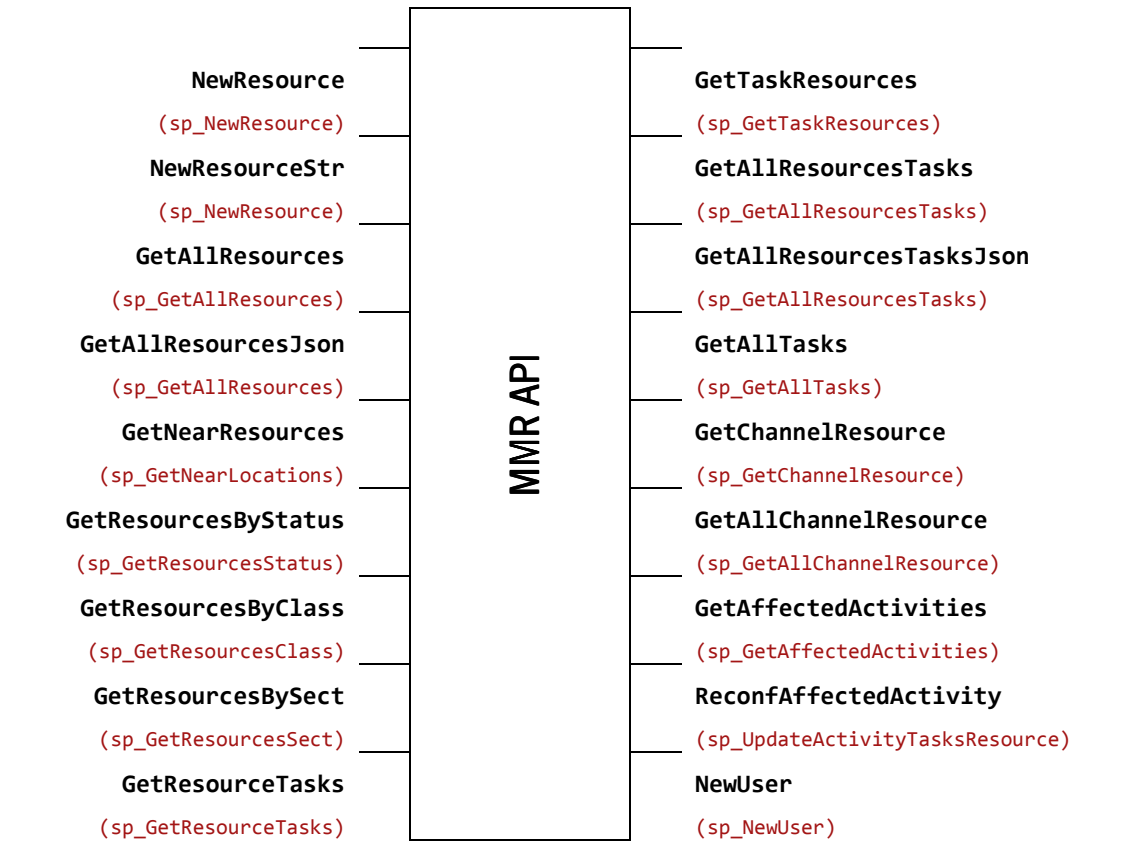


Figure 7.7 - MMR WCF Services API

There was careful to explore the essential activities related to: a) resource management (create and change); b) the selection of resources (sector, status, classification and location); c) management activities (affect a resource) and e) to support the Pragmatic Engine (to manage the communication channels).

These and other methods will be essential for all other services that are intended to implement on the MMR, including Brokering.

7.1.4.2 B – Brokering

In essence the Brokering represents the set of operations responsible for mediation, monitoring and selection of resources.

Monitoring means the operation that allows to know, every instant, everything about a resource, what it does, what it did, what it can do, its state or classification, etc.

Mediation means the behavior that enables the system to interact (remotely) with the resource. Communication channels are an example of that.

Selecting means the operation that allows finding resources in market with greater or lesser rigor of search criteria.

As the resource is a non-entity of MMR, the brokering just "controls" that information that it can "see". The remaining information is the responsibility of the owner of the resource. That is why the task of managing a resource becomes complex.

Therefore, for simplicity, the MMR only changes the state of the resource after any activity he performed or considering the feedback obtained from any other source (customer, supplier, etc.)

Being a resource an independent entity, while “participating” in the MMR nothing prevents it from participating in any type of system. For example, a cutting machine (resource) can perform a task internally and part of his time be allocated for sub-contracting. So, every resource necessarily have a set of complementary information (agenda, for example) important for future developments of this broker.

The API designed to support the Brokering is supported by *Get** methods presented in Figure 7.7. Summarizing, the Broker is responsible by: a) finding a set of resources best suited for a particular task; b) finding a set of alternative resources where the state of some resource so requires; and c) to alert the user to the need for reconfiguration of planned resources.

In the developed component it is possible to select resources by its location, by its state, by its classification and by the operations that it can perform. During the presentation of the application demonstrator, we may see some of the results of the brokering process.

7.1.4.3 C – Pragmatics Engine

The component responsible to support the pragmatics - *Pragmatic Engine*, accounts for all activities associated with the interaction between the interested person and the owner of the resource (provider).

Pragmatics requires that two users can participate in the system in a natural way, i.e., talking face-to-face with the other(s), at that time, being the system only the medium (tool) for this to be possible.

The architecture sought to ensure pragmatic on the system in two ways: a) mediating existing communication channels and b) implementing new channels of communication. Communicational channels mean video chat, video conference, audio chat, email and several others.

In mediation, the system allows each resource owner to appeal the registration and publication of the communicative channel that has and wants to make public (Figure 7.8). Although their state (on, off, faulted, etc.) is moderated by the owner, the system offers the user the possibility to use them whenever they are available.

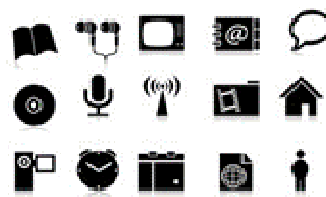


Figure 7.8 - Communicational Channels

Thus the system “finds” the active channels resource and offers them. In practice, analyzing the e-mail service, for instance, the application prepares the message to send and dispatches it to the daemon service responsible for send it. If you need to connect video (video streaming) with the owner of the resource, this communicational channel is established.

The mediation service will be the most suitable, and he "forwards" the process that establishes the communication channel. Thus it can abstract the delicate issues of application integration.

Being a component that operates primarily at the level of the Presentation Layer, the architecture supports the *Pragmatic Engine* through a JQuery library (*pragmatics.js*), developed for this purpose, where a set of methods, via *Controller* and *Model* (MVC architecture), can to get from MMR all the necessary information about resources.

The library would have to respect the independence of the platform (browsers and operating systems) and thus ensure the portability (implemented in Javascript, mainly), so as to be prepared for the scalability needed to function as a cloud-based service. Thus we used the openTok API from TokBox Inc.

"OpenTok is a flexible cloud-based API that makes it easy to add face-to-face video to your application without having to worry about infrastructure, scale, or the latest face-to-face video technology."

openTok, <http://www.tokbox.com/opentok/api/features>

This API is ready for multiple communication models (Table 7.10) and the scalability, supported by cloud, ensures sessions (rooms) between two (P2P rooms) or more. It is easily integrable because it has a Javascript API well documented. In our prototype were integrated and explored video channels over P2P openTok (Private Chats).

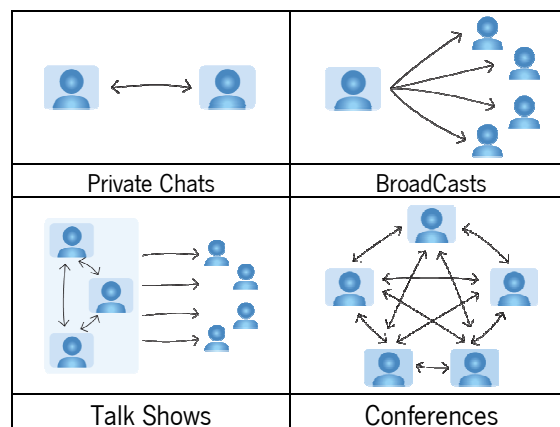


Table 7.10 - openTok Communication Models

(<http://www.tokbox.com/opentok/api/features>)

7.1.5 Web Portal

This web application wants to demonstrate the easy use of MMR API, and thus explore the main capabilities that the architecture enables and promotes.

It is assumed that there is an instance of Manufacturing Market of Resources (MMR) hosted in the cloud. Any interaction with it is done via the API it provides. MMR is operational for several other applications. The Web Portal is just one of them.

For simplification, it was decided to "join" on this website two profiles: a) the customer profile (registered or not) and the profile of the system manager. In practice, offers *back-office* services (to the Manager) and *front-office* for the normal user.

7.1.5.1 Resource

In the *back-office* it is possible to analyze information about any resource, from the specification, state and location (Figure 7.9), the available communication channels (Figure 7.10), the task that can execute, (Figure 7.11), and others.

Show 10 entries	Search:				
Resource	Latitude	Longitude	Classification	State	Sector
Daewoo DMV4020	41.95026	-8.33479	5	On	3
Kulicke & Soffa #1484	41.90000	-8.40000	Sufficient	-1	4
Kulicke & Soffa # 1488	41.90000	-8.50000	Sufficient	On	4
Dimension 3D Printer	41.90008	-8.29311	Good	Busy	2
GUNNAR CUTTING SYSTEM 3001-M	41.90546	-8.29236	Good	On	1
Schaublin Turret	41.77145	-8.44572	Sufficient	Busy	1
pro-Edge For Granite Fabrication	41.80770	-8.86100	Bad	On	2
Hyster yale forklift	41.72160	-8.80750	Sufficient	On	4
Micro Vu Digital Measuring Machine	41.78720	-8.57130	10	On	5
Modular Clean Room	41.85000	-8.28000	Efficient	-1	6
Showing 1 to 10 of 11 entries					

Figure 7.9 - Resources

Show	10	▼	entries	Search: <input type="text"/>														
	Resource	Latitude	Longitude	Classification	State	Sector												
+	Weigh Right Model PMB-4	41.86000	-8.19000	10	On	4												
-	Schaublin Turret	41.77145	-8.44572	Sufficient	Busy	1												
<table><tr><th>Type</th><th>Channel</th></tr><tr><td>Audio</td><td>Audio</td></tr><tr><td>Im</td><td>Messenger</td></tr><tr><td>Email</td><td>xxxx@yyyy.com</td></tr><tr><td>Sms</td><td>989898981</td></tr><tr><td>Video</td><td>Video</td></tr></table>							Type	Channel	Audio	Audio	Im	Messenger	Email	xxxx@yyyy.com	Sms	989898981	Video	Video
Type	Channel																	
Audio	Audio																	
Im	Messenger																	
Email	xxxx@yyyy.com																	
Sms	989898981																	
Video	Video																	
+	pro-Edge For Granite Fabrication	41.80770	-8.86100	Bad	On	2												
+	Modular Clean Room	41.85000	-8.28000	Efficient	-1	6												

<

Figure 7.10 - Resource Communication Channels

Show 10	▼	entries	Search:				
		Resource	Latitude	Longitude	Classification	State	Sector
⊕		Modular Clean Room	41.85000	-8.28000	Efficient	-1	6
⊖		Micro Vu Digital Measuring Machine	41.78720	-8.57130	10	On	5
		Task	Preparation	Execution	Pos-time		
		Band	15:00:00	15:00:00	00:00:00		
		Truss	10:00:00	05:00:00	10:10:00		
⊕		Kulicke & Soffa #1484	41.90000	-8.40000	Sufficient	-1	4
⊕		Kulicke & Soffa # 1488	41.90000	-8.50000	Sufficient	On	4
⊕		Hyster yale forklift	41.72160	-8.80750	Sufficient	On	4
⊕		GUNNAR CUTTING SYSTEM 3001-M	41.90546	-8.29236	Good	On	1
⊕		Dimension 3D Printer	41.90008	-8.29311	Good	Busy	2

Showing 1 to 10 of 11 entries

Figure 7.11 - Resources Tasks

7.1.5.2 Broker

It is also possible to analyze geographically the distribution of resources as well as apply filters on them (brokering).

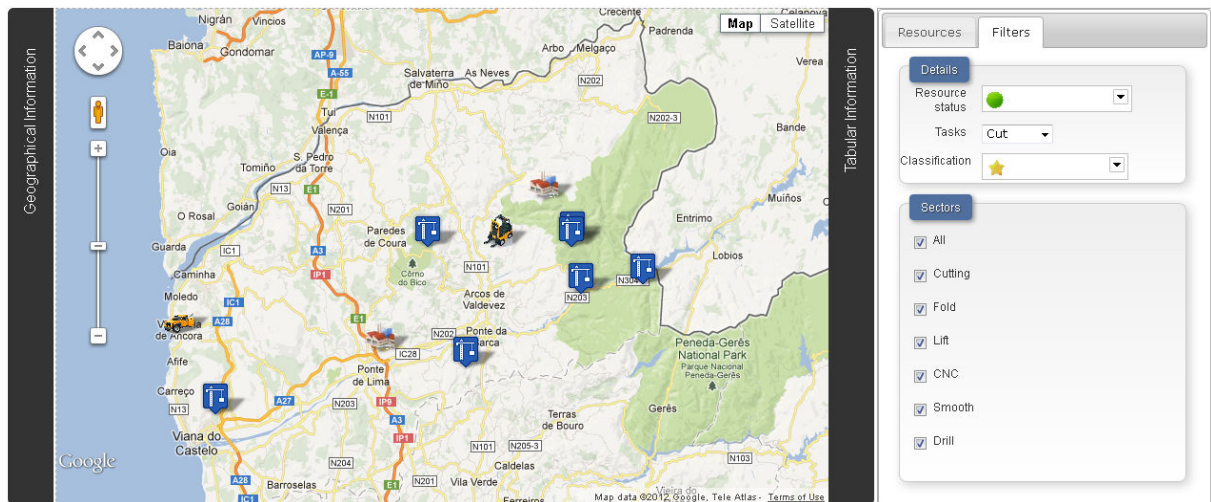


Figure 7.12 - Filtering Resource

There are filters over Resource information (Name, Localization, and Type) (Figure 7.13 (a)), Resource Sectors (Figure 7.13 (b)) and Details (Classification, Status, Tasks) (Figure 7.13 (c))



Figure 7.13 - Resource Filters: (a) General; (b) Sectors; (c) Details

There is also the possibility to find resources using the distance criterion. After selected a particular resource it is possible to identify resources that are distant of a particular distance (20 and 100km, in example of Figure 7.15)

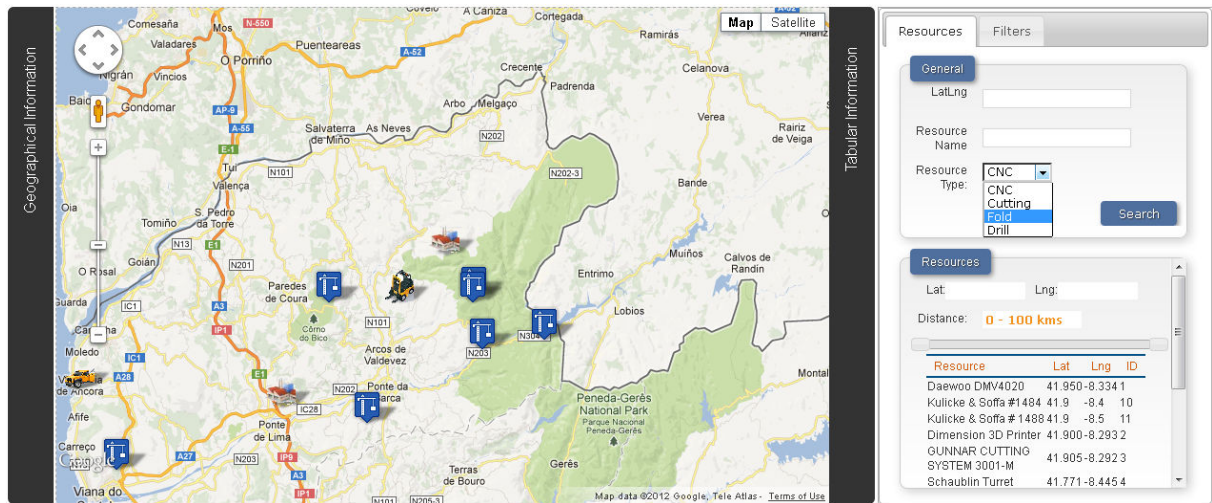


Figure 7.14 - Distance and Others selection criteria

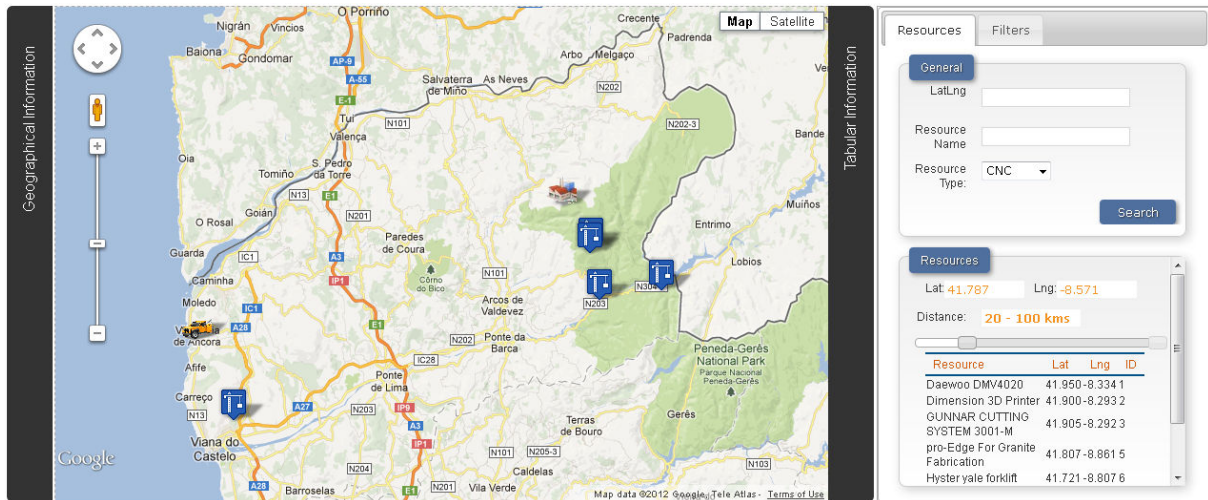


Figure 7.15 - Applied Distance Filter

It is also possible to analyze in detail the resource complementary information, including its status and classification (Figure 7.16 (a)) and their communications channels (Figure 7.16 (b))



Figure 7.16 - Detailed Resource information

Once known the available communications channels, it is possible to use them to immediately contact with the owner of the resource (Figure 7.17).

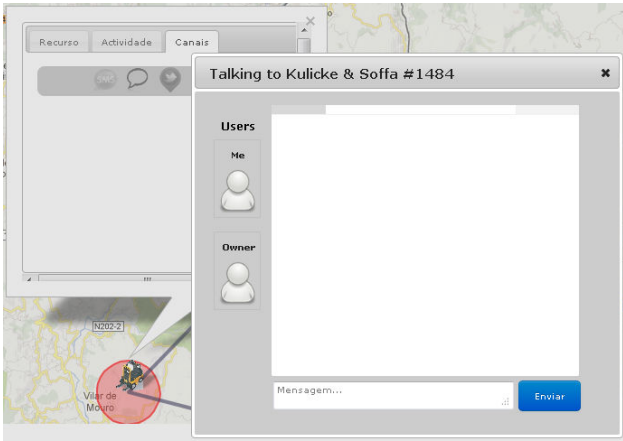


Figure 7.17 - Web Chat with the owner of the resource

7.1.5.3 Dynamic Reconfiguration

For a customer who is registered in the system, his participation is related with the production activities that was schedule to it. You can keep track of their execution or to react to changes of state of some of the involved resources.

It has a geographical distribution of resources involved in the process that will hold, where each step is properly identified by the sequential order that will be executed. May have a purely descriptive perspective (using a table) where the important information about each resource is available (Figure 7.18), or can manage its activity across the map (Figure 7.19).

Planned Activities

Show 10 entries

Steps	Task	Resource	Symbol	State	Source
1	Cut face A	Weigh Right Model PMB-4			A
2	Stitch two faces	Kulicke & Soffa #1484			D
3	Drill 12cm	Daewoo DMV4020			B

Map

Figure 7.18 - Sequence of Production Activities

The state of the resource is highlighted in the table or on a map. In the case where the resource is in an unsatisfactory state (red color), the user can immediately reconfigure his business activity, selecting an alternative resource.

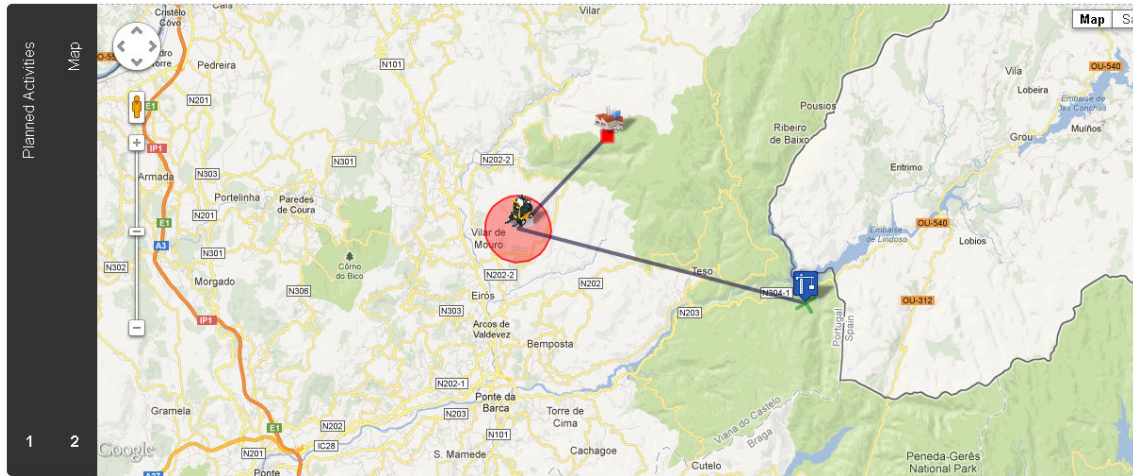


Figure 7.19 - Geographical representation of involved resources

The selection of an alternative resource involves a set of brokering services described below.

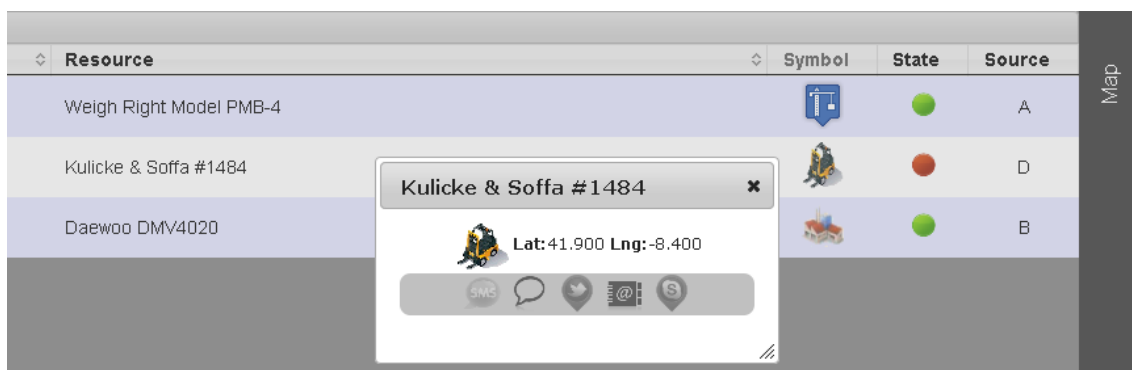


Figure 7.20 - Any resource has their own communication channels

The client may immediately attempt to contact the responsible for the resource (Figure 7.20). Can also check if there are alternative resources (Figure 7.21) and analyze them by their state, their ranks, etc. The contact with the owner of the resource can be done using the channels provided for this purpose (Figure 7.22).

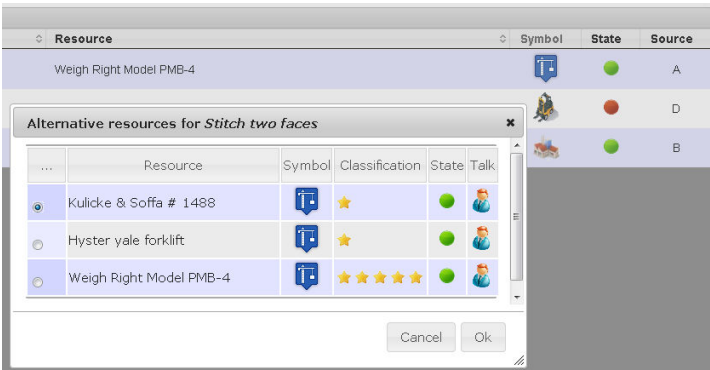


Figure 7.21 - Alternative resources

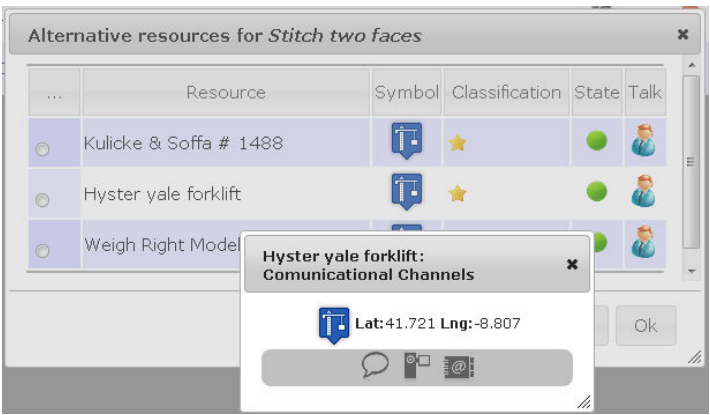


Figure 7.22 - Communication Channels for alternative resource

If it is preferable to analyze the activity via geographic information, the map shows all the steps, from the initial task (red square icon) to the last one (green cross icon) (Figure 7.19).

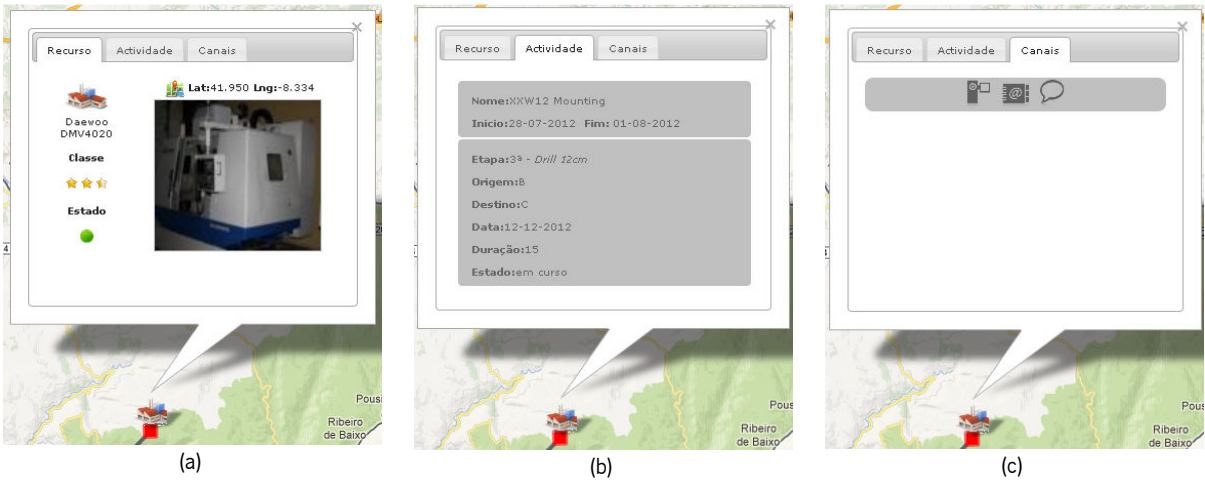


Figure 7.23 - InfoWindow for Resource: (a) General; (b) Occupation and (c) Communicational Channels

In either case the user can easily see all the information about each resource displayed on the map. The information will be displayed on the event *on-mouse-over* when the pointer is over the icon resource in analysis (Figure 7.23).

7.1.6 Mobile Application

To explore the profile of a customer or the conventional resource promoter, an application to be supported by mobile devices was developed. The idea is to demonstrate the easy applicability of MMR API for this type of applications, ensuring the satisfaction of the *multimodality* requirement.

It was demonstrated the use of the system by the promoter of resources on the task of registering and promoting a resource. Once again we tried to simplify and demonstrate just the possibility to interact with the market autonomously and in different contexts.

In practice these are examples of processes that ensure the sustainability of the MMR, allowing the registration and management of each resource, independently of any other application. It is then possible to register a new resource with the information considered essential, including geographical information and communication channels.



Figure 7.24 - Resources registration on the mobile application

This application (for *Windows Phone*) (Figure 7.24) following the MVVM pattern (Freeman & Sanderson, 2011), behaves also as a tool for MMR access that is hosted in the cloud. This

application model will enable to support all services that the Web Portal enables, being the context distinguished by the user profile that is using it.

7.1.7 Data Integration using *Flat File*

In order to integrate with existing applications, the MMR also allows the integration (import) of data according to a specific XML Schema, of which the XML document of Table 7.11 represents an instance. Since we are in the presence of resources with geographic information, used for their representation on a map, the proposed schema should in future include the standard *KML- Keyhole Markup Language* from Google⁴⁵.

Resource XML Flat file
<pre><?xml version="1.0" encoding="utf-8"?> <Resources> <Resource> <id>A0012</id> <name>CNC</name> <address>Arcos</address> <lat>41.95026</lat> <lgt>-8.33479</lgt> <owner><first>lufer</first></owner> <data> <val type="video">http://www.ipca.pt/video</val> <val type="audio">http://www.ipca.pt/audio</val> <val type="chat">gonlufer</val> </data> </Resource> </Resources></pre>

Table 7.11 - Resource XML File

This process allows, for example, the asynchronous batch importation of large amounts of data.

⁴⁵ <https://developers.google.com/kml/documentation/kmlreference>

7.1.8 Repositories

Since it has structured an architecture for dealing with cloud-computing, the amount of simultaneous users to participate in the system is not linear. Continuous access to a BD hosted in the cloud may compromise the performance of the entire system.

Therefore we use repositories that are nothing more than local data structures, *in-memory domain object collection* of the *Repository Pattern* (Fowler, 2002) (Figure 7.25), used to maintain temporary information of the market of resources. These repositories have a particular structure that aims to contribute to the performance of the system.

In addition, the use of repositories represents a layer of security to direct access to the data of the domain (BD). Mediating any access attempt, avoids divulging details on how the BD is implemented.

Usually synchronous accesses are made on these repositories, being possible the direct access to MMR only in extraordinary situations, such as the case of the system startup or data synchronization.

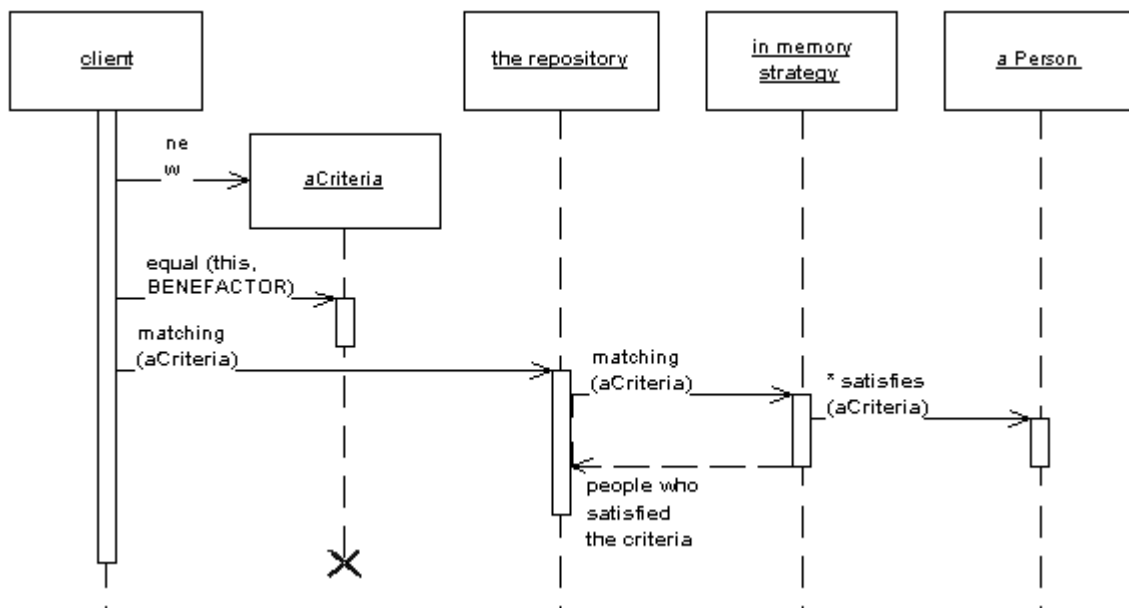


Figure 7.25 - Repository Pattern
(Fowler, 2002)

However, it is essential to maintain repositories updated (in real-time, if possible) and so, parallel and asynchronous operations (using threads or signalR, for example) are, from time to time, responsible to ensure that. This way does not affect the performance of the system and represents an extension to the *Repository Pattern*.

Here is the presentation of the main repositories used in our architecture.

ResourcesRepository

It is the repository to deal with MMR resources (Figure 7.26).

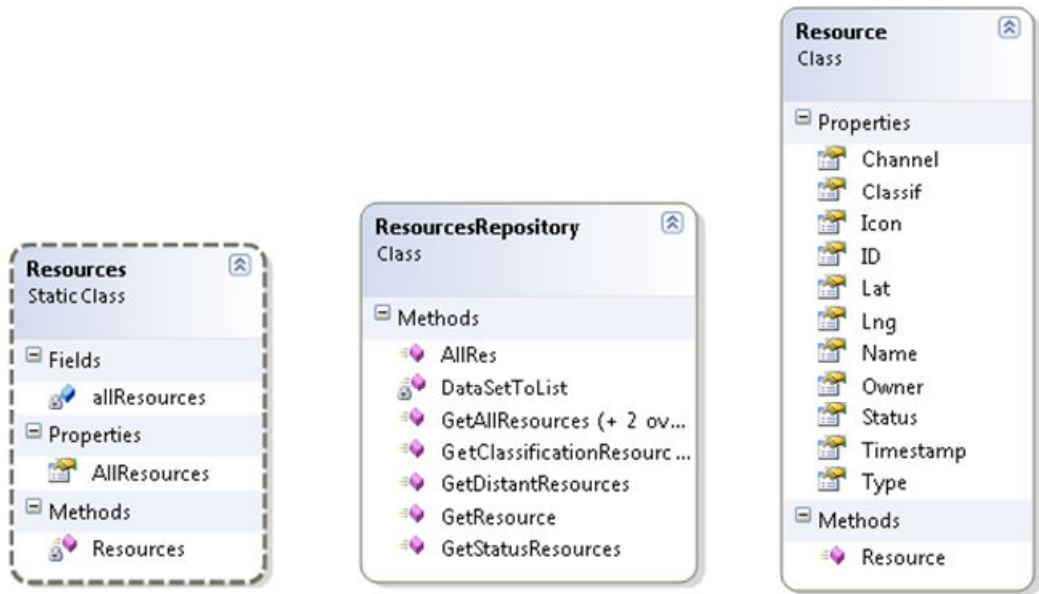


Figure 7.26 - Resources Repository Classes

ChannelRepository

It is the repository to deal with communication channels (Figure 7.27).

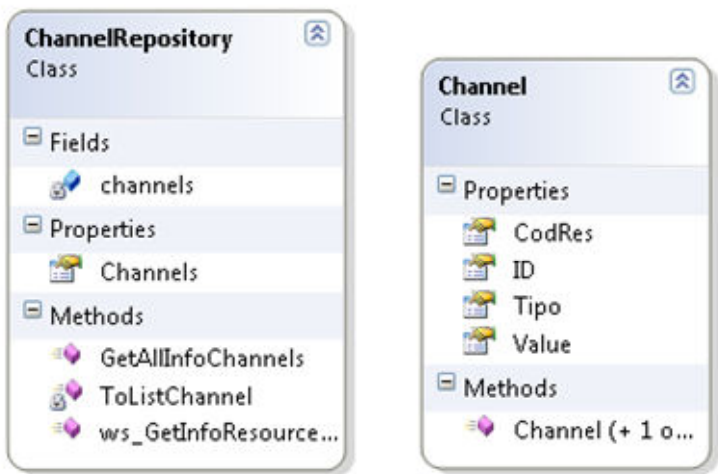


Figure 7.27 - ChannelRepository Classes

TasksResourcesRepository

It is the repository to deal with the resource tasks (Figure 7.28).

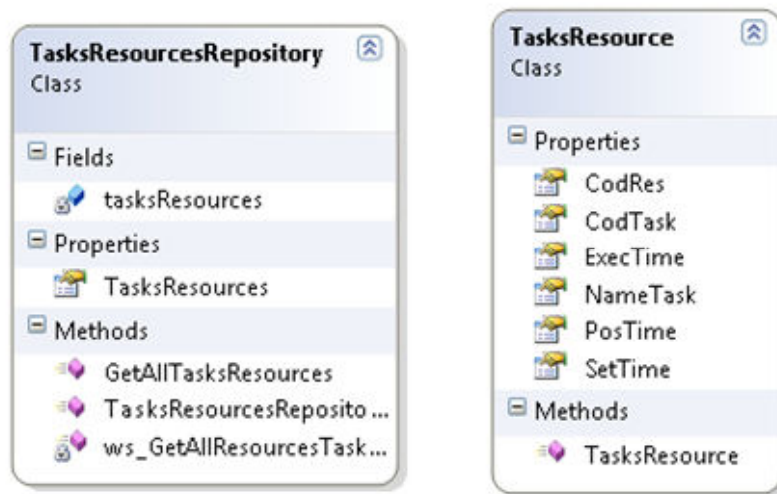


Figure 7.28 - TasksResourcesRepository Classes

7.1.9 Data Serialization

Since have been implemented WCF web services, it is all important that the serialization of data between client and server would be possible using XML (SOAP) or even JSON (read "json"). The first to respond to portability challenges and the second the performance challenges.

LINQ2SQL

```
public JsonResult MorGetActivitiesRepositoryJson(int codUser)
{
    ActivitiesRepository.GetAffectedActivitiesFromBD(codUser);
    var res = ActivitiesRepository.AllActivities();

    var result = from c in res select new object[] {
        c.Seq, c.NameTask, c.NameRes, c.IconPath, c.StaRes,
        c.Source, c.CodRes, c.SecRes, c.NameActi, c.CodActi,
        c.CodTask, c.Lat, c.Lng, c.ImagePath, c.claRes, c.Date,
        c.Destiny, c.Duration, c.StatActi, c.InicioActi, c.FimActi
    };
    return Json(result, JsonRequestBehavior.AllowGet);
}
```

Table 7.12 - LINQ2SQL over XML data

MMR has so WCF services with both capabilities. For example (see the API in Figure 7.7), the **GetAllResourcesJson** method handles JSON data while **GetAllResources** performs the same services but dealing with data structured in XML (*DataSet*).

This particularity is important, for example, during the processing of data for viewing on presentation layer. The example of Table 7.12 shows the generation of JSON applying *LINQ2SQL* on data structured in XML.

7.2 Cirrus - Dynamic Tourism Service

Like we previously explored Manufacturing business activity, a Tourism Activity can be characterized as being complex, with several stakeholders and with precise objectives, but susceptible to easy variations due to “internal” factors, such as tourist interests, or “external” factors, such as weather, economic factors, legislation and others.

Objectively it is intended to apply to the process of definition and preparation of tourist activities, its development and its evaluation. Traditionally the participants in this process are:

- the Tourist, customer that demands and enjoys the activity;
- the Tourist Services Promotor (TSP), which represents a company that offers tourist activities or just a person who rents, for example, a House;
- the Tourism Agent, who intermediates the promoter and the client;
- the *Complementary Service Provider* (CSP) which provides services such as taxi, cicerone, interpret, etc.

All the behavior of the solution revolves around four entities: *Tourist Resource* (TR), *Tourist Operation* (TO), *Tourism Activity* (TA) and *Tourism Market of Resource* (TMR):

- A TR represents an entity able to run or support a task (or operation), in this case, of tourism (TO). Can be a plane, a hotel, a swimming pool, a Museum, a cinema, etc. Can itself be a Tourist Activity (TA) as a whole.
- A TA corresponds to a set of small tasks (TO) that will be performed in certain resources over a period of time. It can be a vacation trip, a visit to a Museum, etc.

- The *Tourism Market of Resources (TMR)* is an instance of a *Market of Resources (MR)* (Cruz-Cunha *et al.*, 2005) and it is intended to provide a repository of resources of interest and a support tool for the management of these resources.

A possible scenario:

"A holiday activity is announced by a travel agency. It is a 15-day trip to the Pure Islands. All the activity is properly detailed day/time and all costs are properly explained. The return and how it will be done is clear too. But it all begins badly. The tourists lost the starting plane and only back to have the plane the next day ..."

Since the aim is not to develop a final application, but rather the demonstration of the applicability of an architecture, and given the complexity and multiplicity of contexts, as happened with other prototype (for manufacturing), we choose to model all entities only with the information considered essential to the services we wanted to demonstrate.

The services that we consider important to analyze to demonstrate the applicability of architecture was:

- a) Tourism Resource and its Communicational Channels Registration
- b) Tourism Operations that Resources can execute
- c) Tourism Activity definition
- d) Searching Resources
- e) Changing the status of the resource
- f) Tourism Activity Reconfiguration and Georeferencing
- g) Tourism Resource Georeferencing
- h) Use of the Communicational Channels

Each Tourist Resource has multiple public information (Figure 7.29) partially managed by the owner of the resource.



Figure 7.29 - Tourist Resource Information: Details; Classification and Communication Channels

There are also three main components in this prototype: a) the *Tourism Market of Resources* with a lot of cloud-based WCF services (Figure 7.7); b) the *Brokering* that, as we described above, represents the set of operations responsible for the mediation, monitoring and selection of tourism resources; and c) the *Pragmatic Engine*, as we also described in Manufacturer prototype, accounts for all activities associated with the interaction between the tourist (client) and the owner of the tourist resource (provider).

It was developed a Web Application to demonstrate the functional part (use and maintenance) of the prototype and a Mobile Application to demonstrate its sustainability. Next we resume some of the main front-ends of these two types of applications (Figure 7.30).

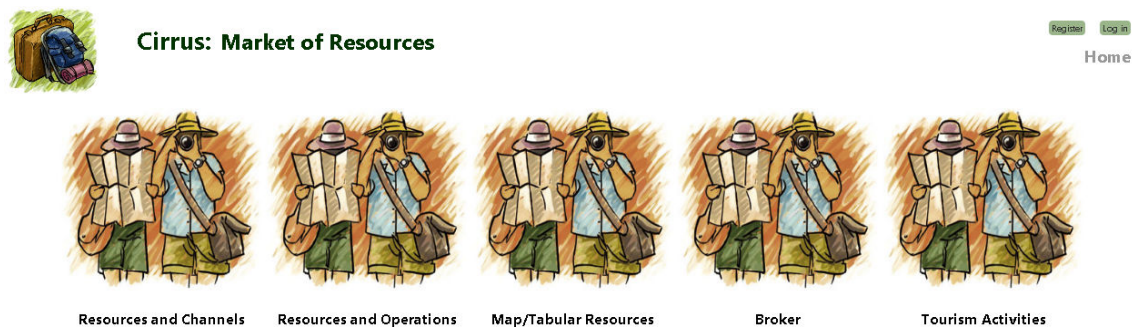


Figure 7.30 - Prototype main Fron-End

Because they are very similar to manufacturing prototype front-ends, we avoid more deep details. They were completely explored in previous prototype description.

A Tourist Resource has detailed information which includes its localization, contacts, operations that it can execute and communicational channels that it supports (Figure 7.31).

Show 10 entries		Search:													
Recurso	Lat	Lng													
Pastores	41.95026	-8.33479													
AAAAA	41.90000	-8.40000													
BBBBB	41.90000	-8.50000													
Urzeira	41.90008	-8.29311													
Cova	41.90546	-8.29236													
Crasto	41.77145	-8.44572													
Ancora	41.80770	-8.86100													
Viana	41.72160	-8.80750													
P. Lima	41.78720	-8.57130													
<table> <tr> <th>Tarefa</th><th>Setup</th><th>Process</th><th>After</th></tr> <tr> <td>Voo</td><td>15:00:00</td><td>15:00:00</td><td>00:00:00</td></tr> <tr> <td>Alojamento T1</td><td>10:00:00</td><td>05:00:00</td><td>10:10:00</td></tr> </table>				Tarefa	Setup	Process	After	Voo	15:00:00	15:00:00	00:00:00	Alojamento T1	10:00:00	05:00:00	10:10:00
Tarefa	Setup	Process	After												
Voo	15:00:00	15:00:00	00:00:00												
Alojamento T1	10:00:00	05:00:00	10:10:00												
Ermelo	41.85000	-8.28000													

Showing 1 to 10 of 11 entries

Figure 7.31 - Tourist Resource Operations

The broker that supports *Brokering* process has several filters that allow the tourist to search, in map or in a table, for a particular kind of resources. Filters by distance, my localization, sector of tourism activity, type of resource, etc., are allowed (Figure 7.32, Figure 7.33).

The interface displays a map of Portugal with various resource markers (hotels, museums, etc.). The sidebar on the right contains the following sections:

- Recursos** (Resources): A tab to view the list of resources.
- Filtros** (Filters): A section for filtering resources.
 - Geral** (General): Includes fields for Lat/Lng, Nome Recurso, and Tipo de Recurso (set to Hotel).
 - Recursos**: A section for filtering by distance, currently set to 30 - 100 kms.

Below the filters, a list of resources is shown:

203A	Hotel Alfa	Porto	65
141R	Museu Serralves	Porto	39
2031	Cinema Avenida	Braga	100

Figure 7.32 - Tourism Brokering services (I)

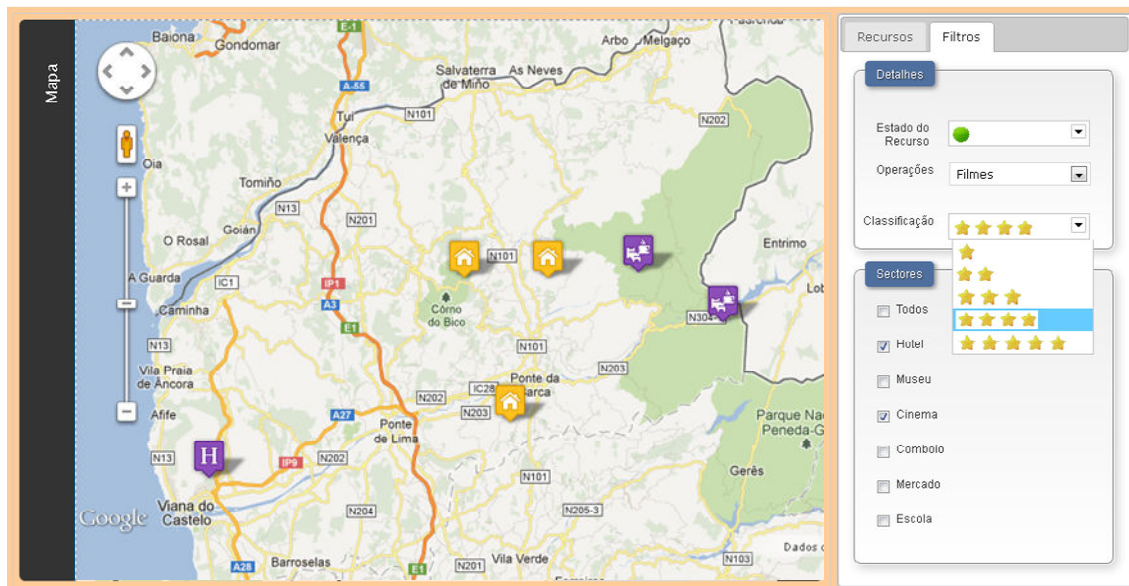


Figure 7.33 - Tourism Brokering services (II)

Since over a map it is possible to get all tourist public resource information: details, classification and communication channels (Figure 7.34).

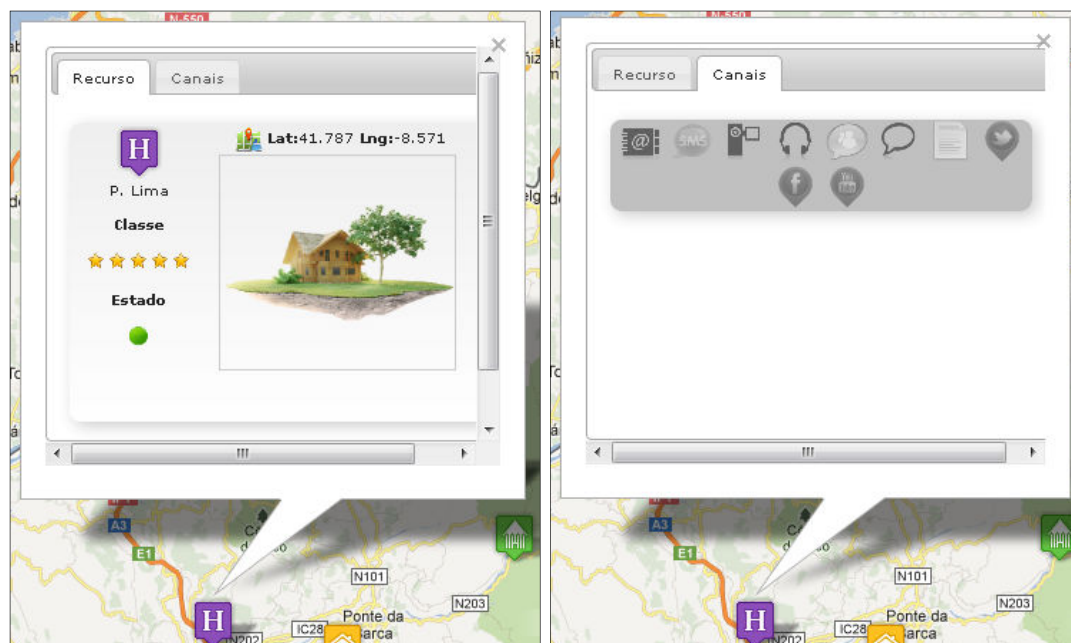


Figure 7.34 - Tourist Resource Public Information

A Tourism Activity can be defined by file (XML file, for instance) and it can be managed using tabular information or using a map.

Resources that can compromise the normal execution of all activity are addressed with a red status in table (Figure 7.35) or a red circle in map (Figure 7.36).

Thus the tourist is advised and can immediately have more details and interact with the owner of that “red resource” (Figure 7.37, Figure 7.38). If needed, the tourist can try to select an alternative resource (Figure 7.39).

Showing 1 to 3 of 3 entries						
Actividades Planeadas	Etapas	Actividade	Recurso	Detalhe	Estado	Origem
	1	Apanhar Avião	AAAAA			A
	2	Ir para Hotel	Viana			D
	3	Alojar no Hotel	P. Lima			B

Figure 7.35 - Tabular Tourism Activity

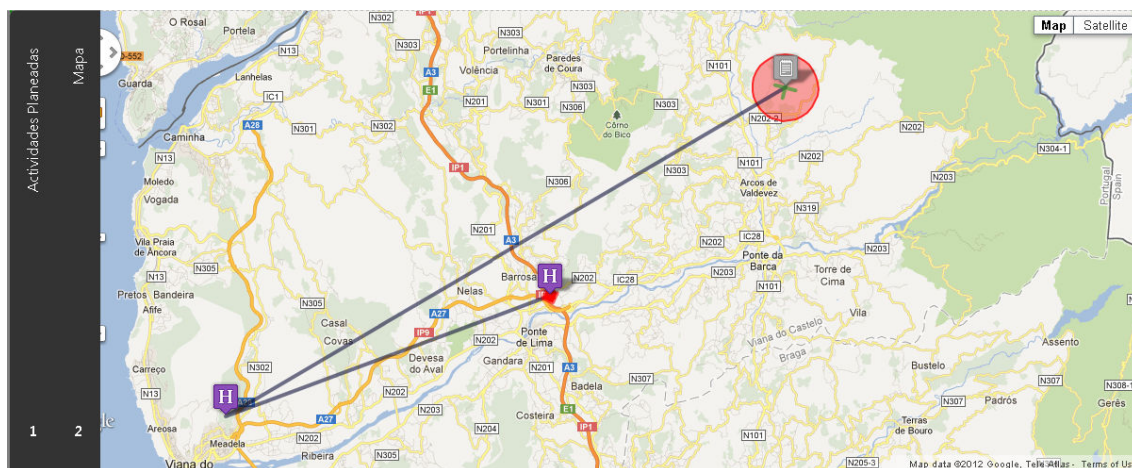


Figure 7.36 - Georeferenced Tourism Activity

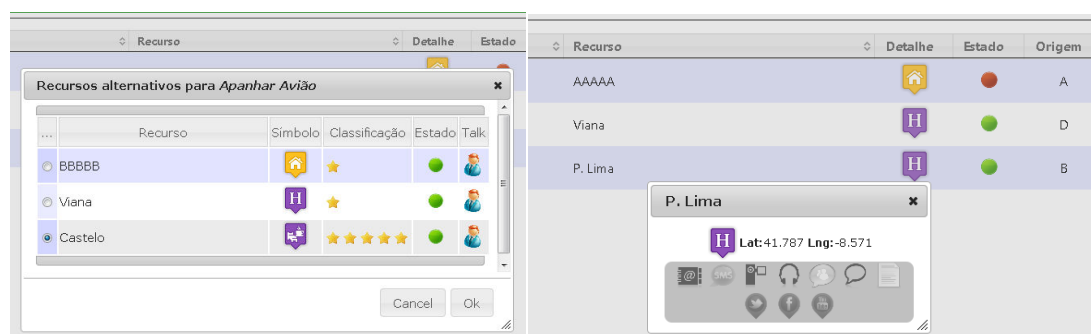


Figure 7.37 - Tourism Activity Reconfiguration



Figure 7.38 - Channel Resources in Tourist Resource Reconfiguration

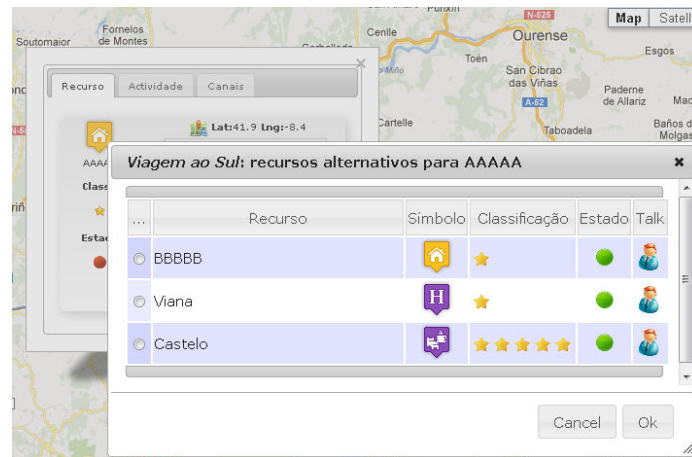


Figure 7.39 - Dynamically alternative Tourist Resources Selection

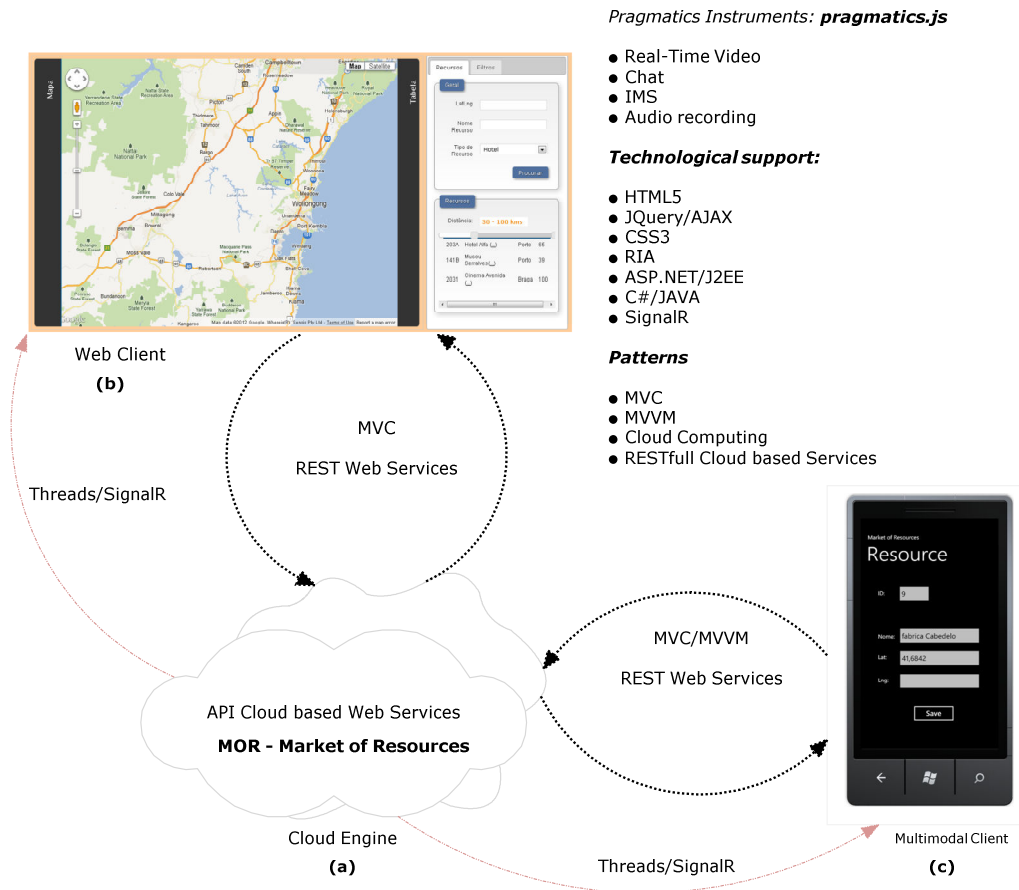
The mobile application is similar to the one described in previous 7.1.6 section.

7.3 Summary

The set of developed solutions demonstrates the applicability of the proposed architecture and the importance of the API that supports the SaaS that manage the Manufacturing or Tourism Market of Resources. From those solutions we can say that:

- i. The Market of Resources (MR) is an autonomous entity whose behavior is independent of the requirements of any application to develop;
- ii. The Manufacturing Market of Resources (MMR) and Tourism Market of Resources (TMR) are supported by (relational) databases hosted in the cloud;
- iii. MMR and TMR having a dedicated API represent a SaaS with a set of cloud services, and makes possible that any platform can fairly easily integrate their services;

- iv. The API allows autonomous registration of resources, communication channels and their management;
- v. Each resource has communication channels provided by its promoter.
- vi. The resource information management is the responsibility of its promoter. The system only monitors that information in the global network of resources.
- vii. A client application needs to manage a set of Repositories to use the information it collects from the MMR or TMR.



- viii. The implementation of Pragmatic Engine through a JQuery library (*pragmatic.js*).
- ix. The client applications will be able to follow the model shown in Figure 7.40:
 - a. MMR and TMR hosted in Cloud
 - b. Web Client with MVC architecture uses MMR or TMR API to be integrated
 - c. Mobile Client with MVC/MVVM architecture uses MMR or TMR API to be integrated
 - d. *Threads* and *SignalR* between clients and MMR or TMR to synchronize data and parallelize operations.

8 CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

This thesis focused on studying the Integration Architectures for Information Systems of Virtual Enterprises on scenarios of their dynamic reconfiguration. It confronts the “traditional” transactional architectures with communicational architectures, arguing that the later are more efficient to ensure effective integration on dynamic reconfiguration scenarios.

The validation of the thesis was sustained through: a) a detailed literature review about Integration Architectures, Patterns and Technologies; Virtual Enterprises and Semiotic Frameworks; and b) a specification of a communicational architecture model. Its validation was obtained through: a) two experimentations on Presentation Layer interoperability using technological mechanisms (ontologies) and b) two Prototypes developed to validate the proposed communicational architecture model.

We address the conclusions drawn from the research presented in this thesis according to the central problem and the research question in section one of this chapter. In section two we detailed the main contributions and finally, in section three, we address the limitations of the research and we make suggestions for further research.

8.1 Conclusions from the research

The central problem of this thesis was:

How to efficiently support and ensure effective integration of information Systems in Virtual Enterprises, in their dynamic reconfiguration scenarios?

Having as main research question:

What is the most appropriate architecture for information systems integration of virtual enterprises that allow effectiveness and efficiency on that integration, in conditions of their dynamic reconfiguration?

And the main hypothesis to demonstrate was:

The architecture of information systems Integration most appropriate in conditions of dynamic reconfiguration of virtual enterprises is the architecture based on communicational systems, contrasting with 'traditional' architectures based on transactional information systems.

Drawing conclusions from the research we can provide the following set of answers to this problem: *First*, we found that purely technological solutions (*transactional architectures*) cannot ensure efficient systems integration. This conclusion arrives from the complexity to develop an efficient integrating solution using existing standards, patterns and architectures. Literature presents several solutions models and patterns but problems still exist. The services quality, for instance, is a relevant problem even considering the most recent technological proposals: SOA and cloud computing. *Second*, we found evidence that the presence of Pragmatics mechanisms to increasing the human-to-human participation on questions analyzes, increased significantly to solve discordances that arises with technological/formal mechanisms and couldn't be managed by them. This conclusion arrives after analyzing the experimentations outcomes. *Third*, we found ample evidence that the existence of a loosely coupled and distributed architecture, following social networks patterns, that means technological independence and services autonomy, are efficient architectures to handle dynamic reconfiguration of Virtual Enterprises. This conclusion comes after explored the developed prototype that explores the architectural proposal. This prototype ensures that: a) the management of resources (*register, change status, delete, etc.*) is the responsibility of their owners; b) the adhesion to the Market of Resources can be made from any platform using appropriated cloud based services; c) the Brokering mechanism of the Market of Resources has sufficient information to efficiently present alternatives resources; and d) faced with multiple alternatives, the user can participate in the final decision about the resource selection, and may even, if needed, start immediately a conversation with the owner of the intended resource. *Fourth*, we found no evidence that transactional architectures are able to support effective integration of information systems. This conclusion comes from two main facts: a) *Transactional architectures* exist to efficiently support systems integration, essentially; b) Effectiveness can only be supported if communicational mechanisms that allow human-to-human co-creation are available. Actually several distinct applications are needed to get (almost) this and they are not integrated, indeed.

8.2 Main Contributions

This thesis made several contributions on the existing theory and research on Integration Systems Architectures to support Dynamic Reconfigurations of Virtual Enterprises:

1. We further developed increasing systems interoperability theory, addressing the relevance of Pragmatics in the construction of effective information systems, towards human-to-human interaction tools.
2. We projected, designed and modeled a communicational architecture as essential complement for traditional transactional architecture to support dynamic reconfiguration of Virtual Enterprises.
3. We projected a Semiotic Framework as the essential base for Pragmatics Technology support.
4. We performed two experimentations on Presentation Layer interpretation, as an experimental mechanism to demonstrate the relevance of Pragmatics in ontologies interoperability and, since, justify the incapacity of only technologies bet. The outcomes of this experimentation provide confidence on initial assumptions and in particular, confirm the hypothesized research framework.
5. We developed two demonstrators as prototypes implementations of proposed communicational architecture, one applied to Manufacturing business activity and Tourism business activity, the other. The outcomes of this prototypes sustain and provide the needed confidence about the following topics:
 - a. The relevance of the existence of distributed platform technological independent towards a ubiquitous information system. The cloud infrastructure and cloud-based services ensure this.
 - b. The relevance of the existence of integrated communication mechanisms to allow human effective and natural participations. The ensured Pragmatics ensures this.

- c. The dynamic reconfiguration inherent to business activities, such as Manufacturing and Tourism, is effectively supported with a communicational architecture with effective brokering mechanisms.
6. We explored emergent and experimental real-time communication technologies (WebRTC and SignalR) to support synchronous/asynchronous communication, and since contributed to cloud computing paradigm.

Achieved all these things above, we believe that:

- a) A reliable framework that can be used to help systems architects and managers to understand the relevance of more than technological aspects, namely social and human ones, to increase the effectiveness of information systems, has being managed, and,
- b) The potential to exploit this in Ubiquitous Cloud Manufacturing and Tourism economic activities is a fact.

8.3 Limitations and suggestions for further research

Analyzing all the steps of this research and confronting the initial objectives, is not difficult to deduce some considerations.

Regarding the technological support of the communicational architecture modeled and prototyped, some technological constraints have limited the intended development.

The area of information technology is evolving at a very fast pace and uncertainty. New technologies arise continuously and with them new possibilities that create new opportunities.

As one of the requirements was intended to project an architecture able to ensure ubiquity, and how the current technology (at the time of the start of work) still does not support it, there was the need to explore new alternatives.

Beyond the learning effort, this process runs into no stable versions of technology, little documented and sometimes too focused on a development plan whose priorities may not coincide with our needs, i.e., the technologies fall short of expected.

This happened with the adoption of cloud infrastructure as a basis for the proposed architecture and the need to explore technologies that ensure real-time communication between services. There were essentially the case of *SignalR* and *WebRTC* (described in Section 3.8.4.2, pag. 62). The limitations of these technologies, browser restrictions, etc., were some of the handicaps that hampered their integration into our architecture, as had been projected.

Regarding the specified communicational architecture, current technologies do not yet ensure that the planned Pragmatic Engine can be properly integrated and operationalized. As technologies that operate on the cloud are still in the development process, this service must still be supported by a number of different applications and hardly integrated. Although the architecture encompasses the definition and transparent use of multiple communication channels (audio, video, text, etc.), the technologies that support them are still too closed and not opened. Technologies such as XMPP or even VOIP still require complex structures to be integrated into current applications under cloud-hosted services model.

With respect to the prototypes developed, virtually none of the modules could be exploited to its full potential. The Brokering and Reconfiguration Manager, could be efficiently implemented but below their potential. The WCF API of cloud-based services is sufficiently robust and complete to demonstrate the main requirements but may include many more services. Security questions were not considered, yet.

The experimentations could only be applied to students in graduation degrees, in order to ensure the necessary profile. The number of students available did not allow that the experimentation could be extended to a larger experimental group. The experimentation did not intend however to focus technological details, but to create the conditions for the application of mechanisms of co-creation between participants, whenever necessary.

In short, the main limitations are focused on technological nature and on the problems that the integration between the heterogeneity of technologies still sustains.

Considering this and projecting future developments, it is important to:

- The development of a markup language – *ADML - Architecture Description Markup Language* capable of modeling a communicational architecture (Anis, 2004).

- Specify a set of effectiveness and efficiency measures and build their data collection instrument for the evaluation of proposed architecture, essential for the desired comparison with other competing architectures;
- Develop new experimentations with performance measures capable of measuring the effectiveness;
- Making reengineering to the specified model for the communicational architecture, and a deeper analysis of available technologies for real-time communication on distributed systems, since they evolve and stabilize.
- To continue the development of prototypes to make them real and useful applications with integrated support of Pragmatics. Communication channels should continue to be explored.
- Reengineering to the technological platform that supports the specified architecture, towards the effective creation of a cloudlets architecture, able to incorporate as services in the new mobile device operating systems.
- Explore simulation platforms as *OpenSimulator* or *SilverLigth*, as well as *Augmented Reality Interfaces*, as potential tools to increase interaction and pragmatics.
- Explore the integration of open source social network engines such as *Elgg* or *Exo*.

As a final note a fact that underlines the importance of this research work. At the time of the start of research, the paradigm of web 2.0 dictated the importance and relevance of the existence of social networks and the potential that multiple applications (isolated) available offered. There were applications for email, chat, video, audio and many others.

After three years we have seen in fact the real confirmation of what we stand for at the beginning of this thesis. Technological advances, conjectures that application integration is essential and those that ensure communication channels are not alien to this process.

In our research were explored mechanisms to ensure this integration. Today are already taken important decisions accordingly. For example, Microsoft acquired Skype project aims to integrate on a single service several services of chat, email and others, which until then were "isolated" in different applications: Messenger, Facebook, Hotmail, Skype and other (Figure 8.1).

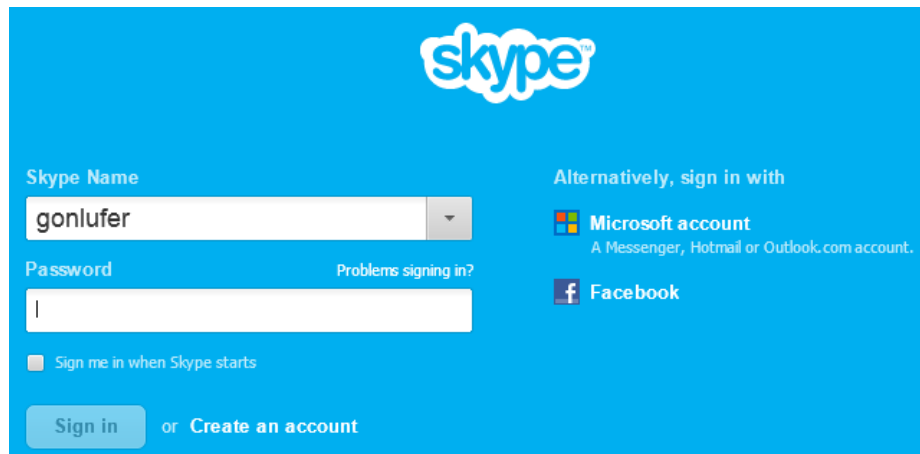


Figure 8.1 - Services Integration

Another current example and interesting is *vSee*⁴⁶. It shows the importance of developing communicational channel video-conference type, with total image sharing (HD video) and devices (screen, disks, etc.), creating, in fact, effective virtual teams (Figure 8.2).

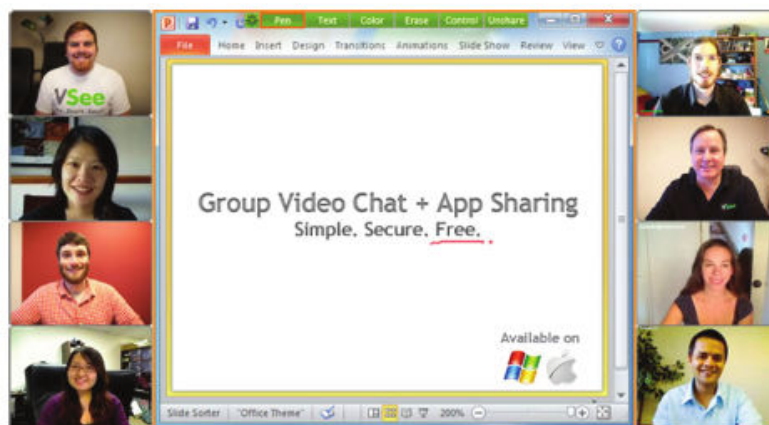


Figure 8.2 - Services integrations and Infra-structures sharing

Completing this work, I hope to have given enough arguments in favor of the use of Pragmatics mechanisms for achieving effective integration in information systems, and sustain that this is a worthwhile field deserving more research.

⁴⁶ <http://vsee.com/>

9 BIBLIOGRAPHY

This chapter presents the author's bibliography as well as the literature research references.

9.1 AUTHOR'S BIBLIOGRAPHY

Ferreira, L., & Putnik, G. D. (2008). Open Tourism Initiative. *Tékhne, VI*.

Ferreira, L., Putnik, G. D., & Cruz-Cunha, M. M. (2010). *Disclosing the Tourism Dynamic Packages*. Paper presented at the CENTERIS 2010 - Conference on Enterprise Information Systems.

Ferreira, L., Putnik, G. D., Cruz-Cunha, M. M., & Putnik, Z. (2012). Towards effective Tourism Dynamic Packages. *Information Resources Management Journal (IRMJ)*, 25(1).

Ferreira, L., Putnik, G. D., Cruz-Cunha, M. M., Putnik, Z., Castro, H., & Alves, C. (2012). *Cloudlet Architecture for Dashboard in Cloud and Ubiquitous Manufacturing*. Paper presented at the CIRP ICME'12 - 8th CIRP Conference on Intelligent Computation in Manufacturing Engineering.

Putnik, G. D., **Ferreira, L.**, Cruz-Cunha, M. M., Putnik, Z., Castro, H., & Shah, V. (2012). User Pragmatics for Effective Ontologies Interoperability for Cloud-based Applications: The case of Ubiquitous Manufacturing System UI. *Electronic Markets – The International Journal on Networked Business*, (submitted for publication).

Putnik, G. D., **Ferreira, L.**, Shah, V., Putnik, Z., Castro, H., Cruz-Cunha, M. M., *et al.* (2011). *Effective Service Dynamic Packages for Ubiquitous Manufacturing System*. In Proceedings of VINORG2011.

9.2 REFERENCES

- Abbaspour, A., & Samadzadegan, F. (2008). Building a context-aware mobile tourist guide system base on a Service oriented architecture. In *Proceedings of The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B4*. Beijing.
- Alptekin, G. I., & Büyüközkan, G. (2011). An integrated case-based reasoning and MCDM system for Web based tourism destination planning. In *Expert Systems with Applications*. Elsevier.
- Ambler, S. (2003). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process* (1. ed.): Wiley.
- Ambler, S. W. (2004). *The Object Primer: Agile Model-Driven Development with UML 2.0* (3. ed.): Cambridge University Press.
- Ambler, S. W. (2011). Communication on Agile Software Projects. *Agile Modeling* Retrieved 4th April, 2012, from <http://www.agilemodeling.com/essays/communication.htm>
- Anis, A. (2004). Composants pour la Modélisation des Processus Métieren Productique, basés sur CIMOSA, *PhD Thesis*. France: Institut Supérieure de Génie Mécanique et Productique.
- Antoniou, G., & Harmelen, F. V. (2008). *A Semantic Web Primer, 2nd edition*. London: The Mit Press.
- ATHENA, C. (2006). ATHENA, Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Applications: FP6-2002-IST-1.
- Bahree, A., Cicoria, S., Mulder, D., Pathak, N., & Peiris, C. (2007). *Pro WCF: Practical Microsoft SOA Implementation*. New York: Apress.
- Barta, R., Feilmayr, C., & Grun, C. (2009). *Covering the Semantic Space of Tourism - An Approach based on Modularized Ontologies*. In Proceedings of 1st Workshop on Context, Information and Ontologies, Heraklion, Greece.
- Bates, B., Freeman, E., Freeman, E., & Sierra, K. (2004). *Head First Design Patterns*. Sebastopol: O'Reilly Media, Inc.
- Begnis, G. (2010). Behavioral Facilitation. Retrieved June, 4, 2012, from <http://encyclopedia.jrank.org/articles/pages/6675/Behavioral-Facilitation.html>
- Berners-Lee. (2008). *The Web Of Things*. ECRIM New 72.
- Berners-Lee, T. (2001). The Semantic Web. *Scientific American* Retrieved 20th May, 2011, from <http://www.sciam.com/article.cfm?id=the-semantic-web>

- Berners-Lee, T., Robert, C., Ari, L., Henrik Frystyk, N., & Arthur, S. (1994). The World-Wide Web. *Communications of the ACM*, 37(8), 76-82.
- Betts, D., Densmore, S., Dunn, R., Narumoto, M., Pace, E., & Woloski, M. (2010). *Developing Applications for the Cloud on the Microsoft® Windows Azure™ Platform*. Microsoft.
- Bhakdi, J. (2010). *Web 3.0 - User-Generated Business, and Why Everyone Becomes a Media Entrepreneur* (Vol. 1.1): SophoMedia.
- Bo-hu, L., Lin, Z., Shi-long, W., Fei, T., Jun-wei, C., Xiao-dan, J., *et al.* (2010). Cloud manufacturing: a new service-oriented networked manufacturing model. *Computer Integrated Manufacturing Systems CIMS 2010*, 16(1), 1-7.
- Booy, D., Liu, K., Qiao, B., & Guy, C. (2008). *A Semiotic Multi-Agent System for Intelligent Building Control*. Paper presented at the Ambi-Sys '08
- Botana, J., King, R., Krämer, T., Rechert, K., Reis, M., Schuster, R., *et al.* (2007). The Planets Interoperability Framework, *An Infrastructure for Digital Preservation Actions*.
- Bremer, M. (2008). Sergei Nirenburg, Victor Raskin, Ontological Semantics. *Minds and Machines*, 18(2), 293-295.
- Brendan, E. (2008, 06). The Messy, Working Web. *Emerging Technologies* Retrieved 12th April, 2012, from http://etech.eweek.com/content/web_technology/the_messy_working_web.html
- Brewster, C. A. (2008). *MIND THE GAP: BRIDGING FROM TEXT TO ONTOLOGICAL KNOWLEDGE*. degree of Doctorate in Philosophy, University of Sheffield.
- Campbell, N. A., Reece, J. B., Taylor, M. R., Simon, E. J., & Dickey, J. L. (2009). In *Biology Concepts & Connections Sixth Edition* (pp. page 2, 3 and G-9): Benjamin Cummings.
- Cardoso, J., & Lange, C. (2007). A Framework for Assessing Strategies and Technologies for Dynamic Packaging Applications in E-Tourism. *Information Technology & Tourism*, 9(1), 27-44.
- Carnap, R. (1942). *Introduction to semantics*. Cambridge: MA: Harvard University Press.
- Carraro, G., & Chong, F. (2006). Software as a Service (SaaS): An Enterprise Perspective: Microsoft.
- Cawood, S., & Fiala, M. (2008). *Augmented Reality: A Practical Guide*: Pragmatic Bookshelf.
- Chaudhuri, S., & Dayal, U. (1997). An overview of data warehousing and OLAP technology. *SIGMOD Rec.*, 26(1), 65-74.
- Chen, D., Dassisti, M., & Tsalgatidou, A. (2005). Interoperability Knowledge Corpus, An intermediate Report: INTEROP NoE.
- Chen, D., Doumeingts, G., & Vernadat, F. (2008). Architectures for enterprise integration and interoperability: Past, present and future. *Computers in Industry*, 59(7), 647-659.

- Cheng, Y., Tao, F., Zhang, L., Zhang, X., Xi, G. H., & Zhao, D. (2010). *Study on the utility model and utility equilibrium of resource service transaction in cloud manufacturing*. Paper presented at the Industrial Engineering and Engineering Management (IEEM), 2010 IEEE International Conference on
- Chiu, D., Yueh, Y., Leung, H.-f., & Hung, P. (2009). Towards ubiquitous tourist service coordination and process integration: A collaborative travel agent system architecture with semantic web services. *Information Systems Frontiers*, 11(3), 241-256.
- Coase, R. (1937). The Nature of the Firm. *Economica*, 4, 386-405.
- Cohen, F. (2008). SOA Virtualization and Composition. *NOW : Enterprise Architecture Principles and Practices*, 12-19.
- Cohen, S. (2007). Ontology and Taxonomy of Services in a Service-Oriented Architecture. *Microsoft Architecture Journal*: Microsoft.
- Coppinger, R. (2007). Cloud computing to enable virtual development. Retrieved 5th June, 2011, from <http://www.flightglobal.com/articles/2010/04/27/340790/cloud-computing-to-enable-virtual-development.html>
- Cordeiro, J., & Filipe, J. (2003). The Semiotic Pentagon Framework, *A perspective on the use of Semiotics within Organizational Semiotics*. Escola Superior de Tecnologia de Setúbal.
- CORDIS. (2011). Future Internet Enterprise Systems Cluster. *European Comissions CORDIS: ICT Research in FP7* Retrieved 3th December, 2011, from http://cordis.europa.eu/fp7/ict/enet/ei_en.html
- Cruz-Cunha, M. M., & Putnik, G. D. (2006). Identification of the domain of opportunities for a market of resources for virtual enterprise integration. *International Journal of Production Research*, 44(12), 2277-2298.
- Cruz-Cunha, M. M., Putnik, G. D., Silva, J., & Santos, J. (2005). Technologies to Support the Market of Resources as an Infrastructure for Agile/Virtual Enterprise Integration. In *Agent and Web Services Technologies in Virtual enterprises*. IGP.
- Cunningham, W. (2003). *Enterprise Solution Patterns Using Microsoft .NET* (Vol. 2.0): Microsoft.
- Daft, R. L., & Lengel, R. H. (1986). Organizational Information Requirements, Media Richness and Structural Design. *Management Science*, 32(5), 554-571.
- Daigneau, R. (2012). *Service Design Patterns - Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services*. Addison-Wesley.
- Davies, J., Studer, R., & Warren, P. (2006). *Semantic Web Technologies: Trends and Research in Ontology-based Systems*. New York: Wiley.
- Deitel, P. J., & Deitel, H. M. (2008). *AJAX, Rich Internet Applications, and Web Development For Programmers*. Prentice Hall.

- Denicolai, S., Cioccarelli, G., & Zucchella, A. (2010). Resource-based local development and networked core-competencies for tourism excellence. *Tourism Management*, 31(2), 260-266.
- Dennis, A. R., & Valacich, J. S. (1999). *Rethinking Media Richness: Towards a Theory of Media Synchronicity*. In Proceedings of 32nd Hawaii International Conference on System Sciences.
- Ding, L., Finin, T., Joshi, A., Peng, Y., Pan, R., Reddivari, P., *et al.* (2005). Search on the Semantic Web. Baltimore: University of Maryland.
- Ding, Y., Fensel, D., Klein, M., & Omelayenko, B. (2002). The semantic web: yet another hip? *Data & Knowledge Engineering*, 41(2-3), 205–227.
- Ding, Y., & Xu, L. (2007). Evolution of the World Wide Web: a historical view and analogical study. *Journal*. Retrieved from <http://www.deg.byu.edu/ding/WebEvolution/evolution-prelude.html>
- Dinh, T. L., & Thi, T. T. P. (2010). A Conceptual Framework for Service Modelling in a Network of Service Systems. In W. Aalst, J. Mylopoulos, N. M. Sadeh, M. J. Shaw & C. Szyperski (Eds.), *Exploring Services Science* (Vol. 53, pp. 192-206). Berlin: Springer Berlin Heidelberg.
- Douglass, B. P. (2002). *Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems*. Addison-Wesley Professional.
- Edmonson, C. (2010). Reveal Responsibilities with Class Diagram. [Video]. *UML08RevealingClassResponsibilities_2MB_ch9.wmv*.
- Elliott, L. (2010). The Business of ICT in Manufacturing in Africa: Afribiz.
- Endrei, M. (2004). *Patterns: Service-Oriented Architecture and Web Services*. In Proceedings of.
- Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, P., *et al.* (2004). *Patterns: Service Oriented Architecture And Web Services*. RedBooks IBM.
- Erl, T. (2005). *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR.
- Erl, T. (2007). *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Alexandria, VA: Prentice Hall.
- Erl, T. (2009). *SOA Design Patterns*. Prentice Hall.
- Erl, T., Karmarkar, A., Walmsley, P., Haas, H., Yalcinalp, L. U., Liu, C. K., *et al.* (2008). *Web Service Contract Design and Versioning for SOA*. Prentice Hall.
- European_Commission. (2002). The Lisbon Strategy - Making Change Happen: Commission of the European Communities.

- European_Commission. (2005). i2010 - A European Information Society for growth and employment: Commission of the European Communities.
- European_Commission. (2009). Europe's Digital Competitiveness Report 2009 Main achievements of the i2010 strategy 2005-2009, *Europe's Digital Competitiveness Report*. European Commission.
- Ferreira, L. (2005). *Formalizing markup languages for user interface*. Master's thesis, Minho University, Braga.
- Ferreira, L., & Putnik, G. D. (2008). Open Tourism Initiative. *Tékhne, VI*.
- Ferreira, L., Putnik, G. D., & Cruz-Cunha, M. M. (2010). *Disclosing the Tourism Dynamic Packages*. Paper presented at the CENTERIS 2010 - Conference on Enterprise Information Systems.
- Ferreira, L., Putnik, G. D., Cruz-Cunha, M. M., & Putnik, Z. (2012). Towards effective Tourism Dynamic Packages. *Information Resources Management Journal (IRMJ)*, 25(1).
- Ferreira, L., Putnik, G. D., Cruz-Cunha, M. M., Putnik, Z., Castro, H., & Alves, C. (2012). *Cloudlet Architecture for Dashboard in Cloud and Ubiquitous Manufacturing*. Paper presented at the CIRP ICME'12 - 8th CIRP Conference on Intelligent Computation in Manufacturing Engineering.
- Figay, N., & Ghodous, P. (2009). Innovative Interoperability Framework for Enterprise Applications within Virtual Enterprise: MEDES 2009.
- FINES (2009). Background. *Journal*. Retrieved from <http://www.fines-cluster.eu/fines/jm/FiNES-Public-Information/background.html>
- Fingar, P. (2009). Extreme Competition, *Cloud Oriented Business Architecture*: Meghan-Kiffer Press.
- Fonseca, F., Câmara, G., & Monteiro, A. M. (2006). A Framework for Measuring the Interoperability of Geo-Ontologies. *Spatial Cognition and Computation*, 6(4), 309-331.
- Fowler, M. (2002). *Patterns of Enterprise Application Architecture* (1. ed.): Addison-Wesley Professional.
- Freeman, A., & Sanderson, S. (2011). *Pro ASP.NET MVC 3 Framework*: Apress.
- Fromkin, V. B. H. S. C., Anna Szabolcsi, Tim Stowell, Donca Steriade. (2000). *Linguistics: An Introduction to Linguistic Theory*. Oxford: Blackwell.
- García, J. G., Calleros, J. M. G., Vanderdonckt, J., & Arteaga, J. M. (2009). A theoretical survey of user interface description languages: IEEE.
- Garrett, J. J. (2010). *The elements of user experience: user-centred design for the web* (2 ed.): New Riders Press.
- Gazendam, H. W. M. (1999). Semiotics, Virtual Organizations and Information Systems.

- Gio, W. (1992). *Mediators in the Architecture of Future Information Systems*.
- Gokhale, A., Kumar, B., & Sahuguet, A. (2002). Reinventing the Wheel? CORBA vs. Web Services. *WWW2002 - Eleventh International World Wide Web Conference* Retrieved 23th, January, 2012, from <http://www2002.org/CDROM/alternate/395/>
- Goldstein, A., Lazaris, L., & Weyl, E. (2011). *HTML5 & CSS3 for the Real World*. SitePoint Pty.
- Group, E. (2010a). *The Future of Cloud Computing: Opportunities for European Cloud Computing behond 2010*. Europe Comission.
- Group, E. (2010b). In K. Jeffery & B. Neidecker-Lutz(Eds.). *The Future of Cloud Computing: Opportunities for European Cloud Computing behond 2010* (Europe Comission.
- GS1. (2008). Mobile Commerce: opportunities and challenges, *A GS1 Mobile Com White Paper*. GS1.
- Guarino, N. (1998). Formal Ontology and Information Systems. In N. Guarino (Ed.), *Proceedings of the First International Conference (FOIS'98)* (pp. 3-15): IOS Press.
- Guarino, N. (2004). Helping people (and machines) understand each other: the role of formal ontology. Trento-Roma, Italy: Laboratory for Applied Ontology (LOA).
- Hansen, M. (2007). *SOA Using Java Web Services*. Prentice Hall.
- Harding, C. (2011). *Cloud Computing for Business*. Van Haren Publishing.
- Harris, D. (2008). *Web 2.0 Evolution into The Intelligent Web 3.0: 100 Most Asked Questions on Transformation, Ubiquitous Connectivity, Network Computing, Open ... Databases and Intelligent Applications*. Emereo Publishing.
- Hartmann, M., & Schreiber, D. (2011). Interacting with Smart Objects. *WorkShop in UiU'11: International Conference on Intelligent User Interfaces*.
- Hasan, A. (2010). Using LINQ in nTier ASP.Net application with JQuery and JSON. 2012, from <http://aspxtutorial.com/post/2010/05/29/VS-2008-Language-Integrated-Query-Architecture-nTier-application-using-LinqJqueryJson-in-aspnet-csharp.aspx>
- Heiner Stuckenschmidt, F. v. H. (2003). *Information Sharing on the Semantic Web* (Vol. PHD): Springer.
- Hill, R., & Wesson, J. (2008). Using Mobile Preference-Based Seraching to Improve Tourist decision Support, *SAICSIT 2008*. ACM.
- Hoekstra, R. (2009). *Ontology Representation: Design Patterns and Ontologies That Make Sense*. IOS Press BV.
- Höhl, W. (2008). *Interactive Environments with Open-Source Software: 3D Walkthroughs and Augmented Reality for Architects with Blender 2.43, DART 3.0 and ARToolKit 2.72* Springer Vienna Architecture.

- Hohpe, G., & Woolf, B. (2003). *Enterprise Integration Patterns*. New York: Addison-wesley Professional.
- Hohpe, G., & Woolf, B. (2004). *Enterprise Integration Patterns : Designing, Building, and Deploying Messaging Solutions*. Boston, MA: Addison-Wesley.
- Hoof, J. (2006). EDA extends SOA and why it is important. *Journal*. Retrieved from <http://soa-eda.blogspot.com/2006/11/how-eda-extends-soa-and-why-it-is.html>
- Höpken, W., Scheuringer, M., Linke, D., & Fuchs, M. (2008). Context-based Adaptation of Ubiquitous Web Applications in Tourism. In P. O'Connor, W. Höpken & U. Gretzel (Eds.), *Information and Communication Technologies in Tourism 2008* (pp. 533-544). New York: Springer.
- Hostetler, G. (2009). *Web Services and SOA Technologies*. O'Reilly.
- Huang, Y., & Bian, L. (2009). A Bayesian network and analytic hierarchy process based personalized recommendations for tourist attractions over the Internet, *Expert Systems with Applications*. Elsevier.
- Hudge (2009). The City of New York Teams Up with HUGE to Launch NYCgo.com. *Journal*. Retrieved from <http://www.hugeinc.com/news/the-city-of-new-york-teams-up-with-huge-and-google-to-launch-nycgo>
- IBM. (2001). Process Broker Services Concepts Guide: WebSphere® Business Integrator.
- IBM. (2006). *Patterns: SOA Client - Access Integration Solutions*. RedBooks.
- IEEE. (1990a). IEEE Standard Computer Dictionary, *A Compilation of IEEE Standard Computer Glossaries*. Institute of Electrical and Electronics Engineers.
- IEEE. (1990b). *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. Institute of Electrical and Electronics Engineers.
- ITIL.org. (2011). ITIL - IT Infrastructure Library. Retrieved 10 October, 2011, from <http://www.itil.org/en/vomkennen/itil/index.php>
- Jaatun, M., Zhao, G., Rong, C., & Zhang, X. (2009). A Semantic Grid Oriented to E-Tourism. In *Cloud Computing* (Vol. 5931, pp. 485-496): Springer Berlin / Heidelberg.
- Jameson, A., & Riedl, J. (2011). Introduction to the Transactions on Interactive Intelligent Systems: ACM Transactions on Interactive Intelligent Systems.
- Jhingran, A. D., Matos, N., & Pirahesh, H. (2010). Information integration: A research agenda. *IBM Systems Journal*
- Johannesson, P., & Perjons, E. (2005). Design Principles for Process Modelling in Enterprise Application Integration: Stockholm University/Royal Institute of Technology.
- Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology Mapping: The State of the Art. *The Knowledge Engineering Review*, 18, 1-31.

- Kashyap, V., & Sheth, A. (1997). *Information Brokering Across Heterogeneous Digital Data: A Metadata-Based Approach*: Kluwer Academic Publishers.
- Kashyap, V., Sheth, A., & Elmagarmid, A. K. (2002). Vocabulary Brokering in the Observer System. *Information Brokering Across Heterogeneous Digital Data, Advances in Database Systems, Volume 20*.
- Katzy, B., & Loh, H. (2003). *Virtual Enterprise Research - State of the Art and Ways Forward*. Munich: CeTIM at University BW.
- Kenteris, M., Gavalas, D., & Economou, D. (2007). *An innovative mobile electronic touris guide application*. London: Springer-Verlag.
- Kim, H. L., Scerri, S., Breslin, J. G., Decker, S., & Kim, H. G. (2008). *The State of the Art in Tag Ontologies: A Semantic Model for Tagging and Folksonomies* In Proceedings of International Conference on Dublin Core and Metadata Applications.
- Knerr, T. (2006). *Tagging Ontology – Towards a Common Ontology for Folksonomies*: Hochschule Furtwangen University.
- Kontchakov, R., Wolter, F., & Zakharyashev, M. (2010). Logic-based ontology comparison and module extraction, with an application to DL-Lite. *Artificial Intelligence*, 174(15), 1093-1141.
- Kuyoro, S. O., Ibikunle, F., & Awodele, O. (2011). Cloud Computing Security Issues and Challenges. *International Journal of Computer Networks (IJCN)*, 3(5), 247-255.
- Lawton, G. (2009). Addressing the Challenge of Cloud-Computing Interoperability. In *Computing Now*. IEEE Computer Society.
- Lee, S., & Cooper, L. F. (2010). Web 2.0 Strives to be Collaboration Software of Choice for Enterprises. Retrieved October, 10th, 2011, from <http://www-01.ibm.com/software/info/itsolutions/collaboration/web20.html>
- Li, M.-S., Cabral, R., Doumeingts, G., & Popplewell, K. (2006). *Enterprise Interoperability Research Roadmap*: Information Society Technologies.
- Li, Y., Chen, H., Zheng, X., Tsai, C.-F., Chen, J.-H., & Shah, N. (2011). A service-oriented travel portal and engineering platform. *Expert Systems with Applications*, 38(2), 1213-1222.
- Linthicum, D. S. (1999). *Enterprise Application Integration*. Addison-Wesley Professional.
- Liu, K. (2008). *Pragmatic Computing - A Semiotic Perspective to Web Services*. In Proceedings of ICETE2007, E-Business and Telecommunications,.
- Liu, S. (2005). *A Theoretic Discussion of Tourism E-commerce: Chongqing Technology and Business University*.
- Lorenzi, F. (2007). A Multiagent Knowledge-Based Recommender Approach with Truth Maintenance, *RecSys'07*: ACM.

- Ma, Y.-T., & Crestan, A. (2009). *Taiwan's challenges for Significant International Tourism Market Growth*. In Proceedings of The Proceedings of the 10th International Digital Government Research Conference.
- Mackie, K. (2007). SOA Synergy and Obstacles. Retrieved March, 13, 2011, from <http://adtmag.com/articles/2007/05/25/soa-synergy-and-obstacles.aspx>
- Maedche, A., & Staab, S. (2001). Learning Ontologies for the Semantic Web. *IEEE Intelligent Systems*, 16(2), 72-79.
- Majumdar, A., & Szigeti, H. (2011). ICT for Manufacturing, *The ActionPlanT Vision for Manufacturing 2.0*. ActionPlanT.
- Malucelli, A. (2006). *Ontology-based Services for Agents Interoperability*. University of Porto, Faculty of Engineering.
- Maréchaux, J.-L. (2006). Combining Service-Oriented Architecture and Event-Driven Architecture using an Enterprise Service Bus. *developerWorks* Retrieved 4th April, 2012, from <http://www.ibm.com/developerworks/library/ws-soa-eda-esb/>
- Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuinness, D., *et al.* (2004). Bringing Semantics to Web Services: The OWL-S Approach. *Welcome to ECS EPrints Repository - ECS EPrints Repository* Retrieved 28 December, 2011, from <http://eprints.ecs.soton.ac.uk/13000/>
- McIlraith, S. A., Son, T. C., & Zeng, H. (2001). Semantic Web services *Intelligent Systems, IEEE*, 16(2), 46-53.
- Mey, J. L. (1993). *Pragmatics: An Introduction*. Oxford: Blackwell
- Meyers, B. C., & Smith, J. D. (2007). Programmatic Interoperability, *Integration of Software-Intensive Systems Initiative*. Carnegie Mellon.
- Mickel, P., Agosto, L., Vignollet, L., & Marty, J.-C. (2006). A Contact Recommender System for a Mediated Social Media. In J. C. Isabel Seruca, Slimane Hammoudi, Joaquim Filipe (Ed.), *Enterprise Information Systems VI*. Springer
- Microsoft. (2002). Applying Microsoft Patterns to Solve EAI Problems.
- Microsoft. (2004). *Guides for Application Integration (Patterns & Practices)*. Microsoft Corporation.
- Microsoft. (2009). *Microsoft Application Architecture Guide: Patterns and Practice*.
- Microsoft, C. (2006, Março). Windows Communication Foundation Architecture Overview. *Microsoft Windows Vista Developer Center* Retrieved 15 April, 2012, from <http://msdn.microsoft.com/en-us/library/aa480210.aspx>
- Moffitt, J. (2010). *Professional XMPP Programming with JavaScript and jQuery*. Wiley Publishing.

- Monod, E. (2004). *Open Tourism: Cultural heritage tourism enhanced by open technologies and u-commerce.*: European Comission - FP6 The Sixth Framework Programme RTD.
- Moor, A. d. (2005). *Patterns for the Pragmatic Web*. In Proceedings of Proc. of the 13th International Conference on Conceptual Structures (ICCS), Berlin.
- Morris, C. (1938). Foundations of the theory of signs. In O. Neurath, R. Carnap & C. Morris (Eds.), *International Encyclopaedia of Unified Science I* (pp. 77–138). Chicago: University of Chicago Press.
- Morris, C. (1995). *Signs, Language and Behavior*. George Braziller.
- Mostafaeipour, A., & Roy, N. (2011). Implementation of Web based Technique into the Intelligent Manufacturing System. *International Journal of Computer Applications*, 17(6).
- Mowbray, T. J., & Zahavi, R. (1995). The essential CORBA: systems integration using distributed objects: John Wiley & Sons,.
- Mulholland, A., Daniels, R., & Hall, T. (2008). The cloud and SOA: Creating an Architecture for today and for the future.
- Najdawi, A. (2009). *SOA and Web Services for Leveraging Inter-organizational Integration in Travel and Tourism* Paper presented at the Services - I, 2009 World Conference.
- Noyes, B. (2001). *WCF RIA Services*. SilverlightShow
- Nunes, L. M. M. (2005). *Learning From Multiple Sources in Heterogeneous Groups of Agents*. Ph.D. tesis, Faculdade de Engenharia da Universidade do Porto.
- Obrst, L. (2004). Ontologies & the Semantic Web for Semantic Interoperability: MITRE - Center for Innovative Computing & Informatics.
- Oliveira, E. (2008). *PatternOnto – Uma proposta metodológica para a integração de padrões de análise e ontologias de domínio*. Universidade Federal de Viçosa, Departamento de Informática.
- Olmedilla, D., & er, M. P. (2005). Interoperability for PeertoPeer Networks: Opening P2P to the rest of the World. Japan: WWW2005.
- Overhage, S. (2002). On Specifying Web Services Using UDDI Improvements, *Net.Objectdays WorkShops*. Germany: Darmstadt University of Technology,.
- Patrizio. (2010). What to Expect from Tech: IT management eBook.
- Paul, S., Pan, J., & Jain, R. (2010). Architectures for the future networks and the next generation Internet: A survey. *Computer Communications*, 34, 2-42.
- Paulheim, H. (2011). *Ontology-based Application Integration*. Springer.
- Paulheim, H., & Probst, F. (2010). Ontology-enhanced user interfaces: a survey. *International Journal on Semantic Web and Information Systems*, 6(2), 36-59.

- Peirce, C. S. (1958). *Collected Papers of Charles Sanders Peirce* (Vol. 1-6).
- Peter, I. (2004). Net History. Retrieved October, 3th, 2011, from <http://www.nethistory.info/History%20of%20the%20Internet/email.html>
- Pettey, C. (2007). Gartner Says 80 Percent of Active Internet Users Will Have A "Second Life" in the Virtual World by the End of 2011, *Gartner. Media Relations*.
- Pokraev, S., & Reichert, M. (2004). Semantic and Pragmatic Interoperability: A Model for Understanding: University of Twente - The Netherlands.
- Prahalad, C. K., & Hamel, G. (1990). The Core Competence of the Corporation. *Harvard Business Review*, 68(3), 79-91.
- Preciado, J. C., Linaje, M., Comai, S., & Sanchez-Figueroa, F. (2007). *Designing Rich Internet Applications with Web Engineering Methodologies*. Paper presented at the Proceedings of the 2007 9th IEEE International Workshop on Web Site Evolution.
- Putnik, G. D. (2005). Lecture Notes on Virtual Enterprises: University of Minho.
- Putnik, G. D. (2010). *Ubiquitous Manufacturing Systems vs. Ubiquitous Manufacturing Systems: Two Paradigms*. In Proceedings of Proceedings of the CIRP ICME '10 - 7th CIRP International Conference on Intelligent Computation in Manufacturing Engineering - Innovative and Cognitive Production Technology and Systems.
- Putnik, G. D., & Cruz-Cunha, M. M. (2005a). *Virtual Enterprise Integration: Technological and Organizational Perspectives*. Hershey, PA, USA: IDEA Group Publishing.
- Putnik, G. D., & Cruz-Cunha, M. M. (2007). *Knowledge and Technology Management in Virtual Organizations*. IDEA Group Publishing.
- Putnik, G. D., Cruz-Cunha, M. M., Sousa, R., & Ávila, P. (2005b). Virtual Enterprise Integration: Challenges of a New Paradigm. In *Virtual Enterprise Integration: Technological and Organizational Perspectives* (pp. 1-30 (ISBN: 31-59140-59405-59143; ISBN: 59141-59140-59406-59141; eISBN: 59141-59140-59407-X)). Hershey, PA, USA: IDEA Group Publishing.
- Putnik, G. D., Ferreira, L., Cruz-Cunha, M. M., Putnik, Z., Castro, H., & Shah, V. (2012). User Pragmatics for Effective Ontologies Interoperability for Cloud-based Applications: The case of Ubiquitous Manufacturing System UI. *Electronic Markets – The International Journal on Networked Business*, (submitted for publication).
- Putnik, G. D., Ferreira, L., Shah, V., Putnik, Z., Castro, H., Cruz-Cunha, M. M., *et al.* (2011). *Effective Service Dynamic Packages for Ubiquitous Manufacturing System*. In Proceedings of VINORG2011.
- Putnik, G. D., & Putnik, Z. (2010a). A semiotic framework for manufacturing systems integration - Part I: Generative integration model. *International Journal of Computer Integrated Manufacturing*, 23: 8, 691 - 709.

- Putnik, G. D., & Putnik, Z. (2010b). A semiotic framework for manufacturing systems integration – Part I: Generative integration model. *International Journal of Computer Integrated Manufacturing*, Vol. 23, Nos. 8–9, August–September 2010, 691-709.
- Raney, D. (2009). The Evolution of Client/Server Computing. Retrieved September, 2th, 2011, from http://cis.cuyamaca.net/draney/214/web_server/client.htm
- Recanati, F. (2004). Pragmatics and semantics In L. R. H. a. G. Ward (Ed.), *The Handbook of Pragmatics*. Oxford, UK: Blackwell Publishing.
- Reese, G. (2009). *Cloud Applications Architectures*. O'reilly.
- Regina, M. M. B., Marta, M., & Claudia, M. L. W. (2001). The use of mediation and ontology technologies for software component information retrieval. *SIGSOFT Softw. Eng. Notes*, 26(3), 19-28.
- Repenning, A., & Sullivan, J. (2003). *The Pragmatic Web: Agent-Based Multimodal Web Interaction with no Browser in Sight*. In Proceedings of IFIP INTERACT03: Human-Computer Interaction, Zurich, Switzerland.
- Saint-Andre, P. (2011). Extensible Messaging and Presence Protocol (XMPP). In R. 6120 (Ed.): Cisco.
- Saussure, F. d. (1916). *Course in general linguistics* (W. Baskin, Trans.): McGraw-Hill Book Company.
- Schiaffino, S., & Amandi, A. (2009). Building an expert travel agent as a software agent, *Expert Systems with Applications*. Elsevier.
- Schoop, M., de Moor, A., & Dietz, J. L. G. (May 2006). The pragmatic web: a manifesto. *Communications of the ACM - Two decades of the language-action perspective*, Volume 49 Issue 45.
- Schultheiss, B. C., Rijn, L. C. J. V., & Kamphuis, C. (2002). Secure Meta-computing in an Extended Enterprise: NLR - National Aerospace Lanoratoty.
- Schwinger, W., Grün, C., Pröll, B., Retschitzegger, W., & Werthner, H. (2006). Pinpointing Tourism Information onto Mobile Maps – A Light-Weight Approach. In M. Hitz, M. Sigala & J. Murphy (Eds.), *Information and Communication Technologies in Tourism 2006* (pp. 29-43). Viena: Springer Computer Science.
- Sevinc, I., & D'Ambra, J. (2004). *EXTENDING MEDIA RICHNESS THEORY: THE INFLUENCE OF A SHARED SOCIAL CONSTRUCTION*. In Proceedings of ECIS.
- Shahzad, S. K. (2011). Ontology-based User Interface Development: User Experience Elements Pattern *Journal of Universal Computer Science*, 17(7), 1078-1088.
- Shannon, C. E., & Weaver, W. (1949). *A Mathematical Model of Communication*. Urbana, IL: University of Illinois Press

- Shen, W., Hao, Q., Wang, S., Li, Y., & Ghenniwa, H. (2007). *An agent-based service-oriented integration architecture for collaborative intelligent manufacturing*. In Proceedings of Robotics and Computer-Integrated Manufacturing.
- Singh, P. (2003). Examining the Society of Mind. *Computing and Informatics*, 22, 521-543.
- Smith, J. (2009). WPF Apps With The Model-View-ViewModel Design Pattern. Retrieved 8th October, 2011, from <http://msdn.microsoft.com/en-us/magazine/dd419663.aspx>
- Sommerville, I. (2000). *Software Engineering*. Addison Wesley.
- Sprott, D., & Wilkes, L. (2004). Understanding Service-Oriented Architecture, *The Architecture Journal*. Microsoft.
- Spyns, P., & Meersman, R. (2007). Ontology Engineering and (Digital) Business Ecosystems: a case for a Pragmatic Web: IEEE.
- Stamper, R. (1999). Organisational semiotics: Informatics without the computer? In R. C. K. Liu, and P.B. Andersen (Ed.), *Information, organisation, and technology: studies in organisational semiotics*. Dordrecht: Kluwer.
- Stuckenschmidt, H., & Harmelen, F. v. (2005). *Information Sharing on the Semantic Web* (Vol. XIX): Springer.
- Sycara, K., Paolucci, M., Ankolekar, A., & Srinivasan, N. (2003). Automated discovery, interaction and composition of Semantic Web services. *Journal of Web Semantics*, 28-32.
- Talend. (2011). Cloud Integration: 4 Key Recommendations: Talend - Open Integration Solutions.
- Tamani, E., & Evripidou, P. (2007). Combining Pragmatics and Intelligence in Semantic Web Service Discovery. In *OTM 2007* (pp. 824–833): Springer-Verlag Berlin Heidelberg.
- Titus, T. (2004). *C# Threading Handbook* APress, LLC.
- Torniai, C., Brush, M., Vasilevsky, N., Segerdell, E., Wilson, M., Johnson, T., *et al.* (2011). *Developing an application ontology for biomedical resource annotation and retrieval: challenges and lessons learned*. In Proceedings of ICBO: International Conference on Biomedical Ontology, Buffalo, NY, USA.
- Tripathi, R., Gupta, M. P., & Bhattacharya, J. (2006). Dimensions of Interoperability for an Effective Portal: Indian Institute of Technology.
- Tudorache, T., Noy, N. F., Falconer, S. M., & Musen, M. A. (2011). *A Knowledge Base Driven User Interface for Collaborative Ontology Development*. Paper presented at the UIU'11: 2011 International Conference on Intelligent User Interfaces.
- Usmani, S., Azeem, N., & Samreen, A. (2011). Dynamic Service Composition in SOA and QoS Related Issues *International Journal of Computer Technology and Applications*, 2, 1315-1321.

- Villela, M. L. B. (2004). *Validação de Diagramas de Classe por meio de Propriedades Ontológicas*. Universidade Federal de Minas Gerais, Belo Horizonte.
- W3C. (2005). Web Service Semantics - WSDL-S.
- W3C. (2011). Widget Packaging and XML Configuration, *W3C Recommendation 27 September 2011*.
- Watson, R., Akselsen, S., Monod, E., & Pitt, L. (2004). The Open Tourism Consortium:: Laying The Foundations for the Future of Tourism. *European Management Journal*, 22(3), 315-326.
- Wiehler, G. (2004). Web Services and Service Oriented Architectures - The Impact on Business Applications: Siemens.
- William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick, & Mowbray, T. J. (1998). *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis* Wiley.
- Williamson, O. (1979). Transaction cost economics: the governance of contractual relations. *Journal of Law and Economics*, 22(2), 233-260.
- Wilson, C. (2008). Avatars, Virtual Reality Technology, and the U.S. Military: Emerging Policy Issues. *Journal*. Retrieved from <http://www.fas.org/sgp/crs/natsec/RS22857.pdf>
- Workgroup, S. (2010). *Service-Oriented Architecture Ontology*. The Open Group.
- Xu, X. (2012). From cloud computing to cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*(28), 75-86.
- Yeh, A., & Nason, R. (2004). Toward a Semiotic Framework for Using Technology in Mathematics Education: The Case of Learning 3D Geometry.
- Yigitbasioglu, O. M., & Velcu, O. (2010). A review of dashboards in performance management: Implications for design and research. *International Journal of Accounting Information Systems*, 13(1), 41-59.
- Yu, L. (2007). *Introduction to the Semantic Web and Semantic Web Services*. Boca Raton: Chapman & Hall/Crc.
- Zhang, X., Song, H., & Huang, G. Q. (2009). Tourism supply chain management: A new research agenda. *Tourism Management*, 30(3), 345-358.
- Zhuk, J. (2004). *Integration-Ready Architecture and Design*. Cambridge University Press.
- Zwegers, A. (2005). The Enterprise Interoperability Cluster - Developing open and secure technologies to connect systems and enterprises. *Journal*. Retrieved from <http://www.ve-forum.org/apps/pub.asp?Q=1284>

10 APPENDIX A – PROTOTYPE DESCRIPTION

Introduction

This appendix presents the services and the respective user interfaces of the Ubiquitous Manufacturing System Prototype, developed over the proposed cloud-based communicational architecture.

The prototype focused services integration and their usability, rather than ergonomic and aspects particularities. It explored emergent web3.0 and cloud computing technologies to support asynchronous and real-time distributed services (Cloud engine, WCF Cloud-based services, Threads and SignalR), as well as user interaction web3.0 requirements (HTML5, CSS3, JQuery). Further work must explore responsive dashboards, multimodal services and user experience in depth.

Cloud-Based Services

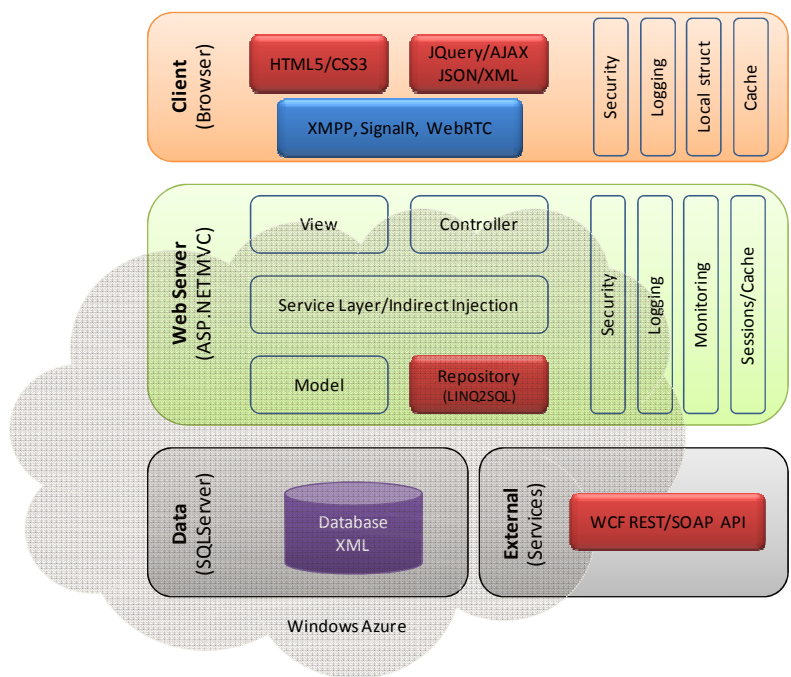


Figure 10.1 - Supporting Platform

The Prototype represents an integrating application that implements the proposed architecture (Figure 10.1), emphasizing the integration of: a) existing cloud-based services, using RESTfull or SOA Services API and b) created Manufacturing Market of Resources's (MMR) cloud-based services, using its own developed API.

Given the complexity of Manufacturing, and considering the architecture applicability as the main goal, we decide to model the entities only with the information considered essential to the services we wanted to demonstrate. There are:

- a) Resource Registration
- b) Communicational Channels Registration
- c) Tasks that Resources can execute
- d) Production Plan definition
- e) Search Resources by state, classification, sector, georeferenced information
- f) Changing the status of the resource
- g) Production Plan Reconfiguration
- h) Resources Georeferencing
- i) Production Plan Georeferencing
- j) Communicational Channels Integration

Dashboard Graphical User Interface

Resource Registration

The Resource registration service represents one of the essential platform services. It allows and assures the autonomous sustainability of resources network. As happens with actual social network, resources are managed by its owner, essentially. The owner promotes or removes it from the resources network.

It is support by a MMR API and can be easily used by any kind of application. In our prototype we explored a mobile Windows Phone application and a bulk loading service using XML flat files.



Figure 10.2 - PhoneClient Resource Registration

Communication Channel Registration



Figure 10.3 - PhoneClient Resource Channels Registration

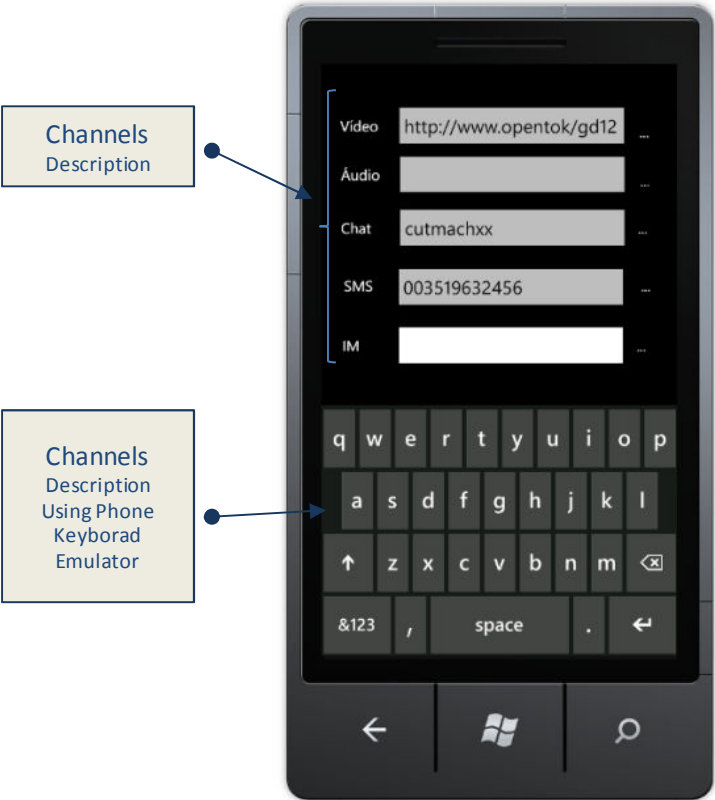


Figure 10.4 - PhoneClient Keyboard Emulator



Figure 10.5 - PhoneClient Market of Resources Management

Bulk Resource Registration

The method *GetAllResourcesFromFile* was created to support mechanisms to import large amount of resource information. XML were the data format explored. However, the method is already prepared to deal with JSON data format. It is implemented with LINQ to XML.

```
void GetAllResourcesFromFile(char type, string fileName);
```

Resource XML Flat file

```
<?xml version="1.0" encoding="utf-8"?>
<Resources>
  <Resource>
    <id>A0012</id>
    <name>CNC</name>
    <address>Arcos</address>
    <lat>41.95026</lat>
    <lgt>-8.33479</lgt>
    <owner><first>lufer</first></owner>
    <data>
      <val type="video">http://www.ipca.pt/video</val>
      <val type="audio">http://www.ipca.pt/audio</val>
      <val type="chat">gonlufer</val>
    </data>
  </Resource>
</Resources>
```

The XML is validated using the specific XML XSchema presented in Table 10.1. Figure 10.6 represents graphically the hierarchy of XML Schema elements.

Resource XML Xschema

```
<?xml version="1.0" encoding="UTF-8"?>
<!--by lufer-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="name" type="xs:string"/>
  <xs:element name="lgt" type="xs:string"/>
```

```

<xs:element name="lat" type="xs:string"/>
<xs:element name="id" type="xs:string"/>
<xs:element name="first" type="xs:string"/>
<xs:element name="second" type="xs:string"/>
<xs:element name="address" type="xs:string"/>
<xs:element name="val">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="type" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:element name="owner">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="first"/>
      <xs:element ref="second"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="data">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="val" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Resources">
  <xs:complexType>
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
      <xs:element ref="Resource"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Resource">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="id"/>
      <xs:element ref="name"/>
      <xs:element ref="address"/>
      <xs:element ref="lat"/>
      <xs:element ref="lgt"/>
      <xs:element ref="owner"/>
      <xs:element ref="data"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>

```

Table 10.1 – XML Schema for XML validation

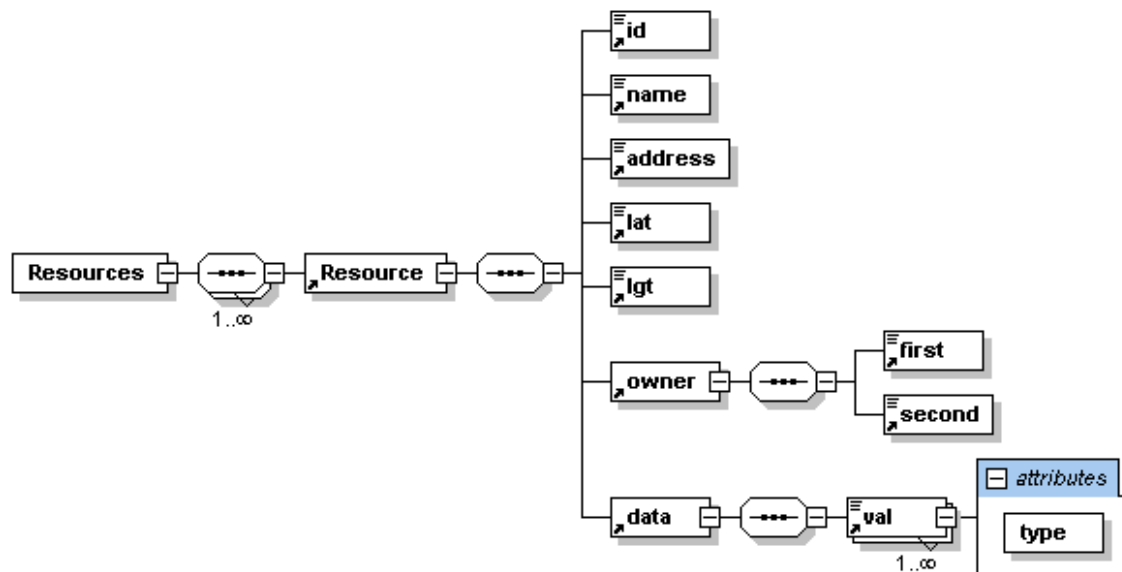


Figure 10.6 - XML Schema for XML validation – Graphical representation

Resource Administration

The main goal of this service was to demonstrate the integration of several mechanisms to analyze existing resources information and their communication channels. All features are supported by MMR cloud-based services and the front-end explores web 3.0 technologies, mainly JQuery, CSS3 and HTML5.

It is possible to analyze all resources data (Figure 10.7), theirs communication channels (Figure 10.8) and the tasks that they can execute (Figure 10.8).

Resource	Latitude	Longitude	Classification	State	Sector
Daewoo DMV4020	41.95026	-8.33479	5	On	3
Kulicke & Soffi		-8.40000	Sufficient	On	4
Kulicke & Soffi		-8.50000	Sufficient	On	4
Dimension 3D Printer		-8.29311	Good	Busy	2
GUNNAR CUTTING SYSTEM 3001-M	41.90546	-8.29236	Good	On	1
Schaublin Turret	41.77145	-8.44572	Sufficient	Busy	1
pro-Edge	41.80770	-8.86100	Bad	On	2
Hyster ya	41.72160	-8.80750	Sufficient	On	4
Micro Vu Digital Measuring Machine	41.78720	-8.57130	10	On	5
Modular Clean Room	41.85000	-8.28000	Efficient	-1	6

Showing 1 to 10 of 11 entries

Figure 10.7 - Resources List

Resource	Latitude	Longitude	Classification	State	Sector
Weigh Right Model PMB-4	41.86000	-8.19000	10	On	4
Schaublin Turret	41.77145	-8.44572	Sufficient	Busy	1
pro-Edge For Granite Fabrication	41.80770	-8.86100	Bad	On	2
Modular Clean Room	41.85000	-8.28000	Efficient	-1	6

Type	Channel
Audio	Audio
Im	Messenger
Email	xxxx@yyyy.com
Sms	989898981
Video	Video

Figure 10.8 - Resources Channels

Resource	Latitude	Longitude	Classification	State	Sector
Modular Clean Room	41.85000	-8.28000	Efficient	-1	6
Micro Vu Digital Measuring Machine	41.78720	-8.57130	10	On	5
Kulicke & Soffa #1484	41.90000	-8.40000	Sufficient	-1	4
Kulicke & Soffa # 1488	41.90000	-8.40000	Sufficient	On	4
Hyster yale forklift	41.90750	-8.29236	Sufficient	On	4
GUNNAR CUTTING SYSTEM 3001-M	41.90546	-8.29236	Good	On	1
Dimension 3D Printer	41.90008	-8.29311	Good	Busy	2

Task	Preparation	Execution	Pos-time
Band	15:00:00	15:00:00	00:00:00
Truss	10:00:00	05:00:00	10:10:00

Figure 10.9 - Resource Tasks

Broker

The Broker is a set of mechanisms which allow resources selection. This component offers filters over Resource information (Name, Localization, Type) (Figure 10.11), Resource Sectors and Details (Classification, Status, Tasks) (Figure 10.12).

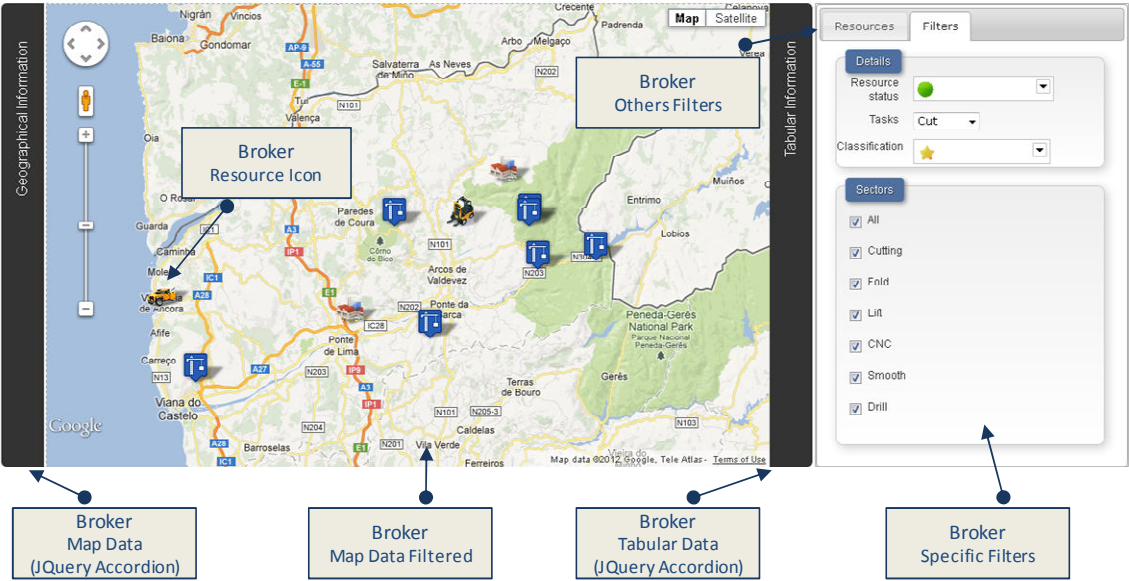


Figure 10.10 - Broker Filters

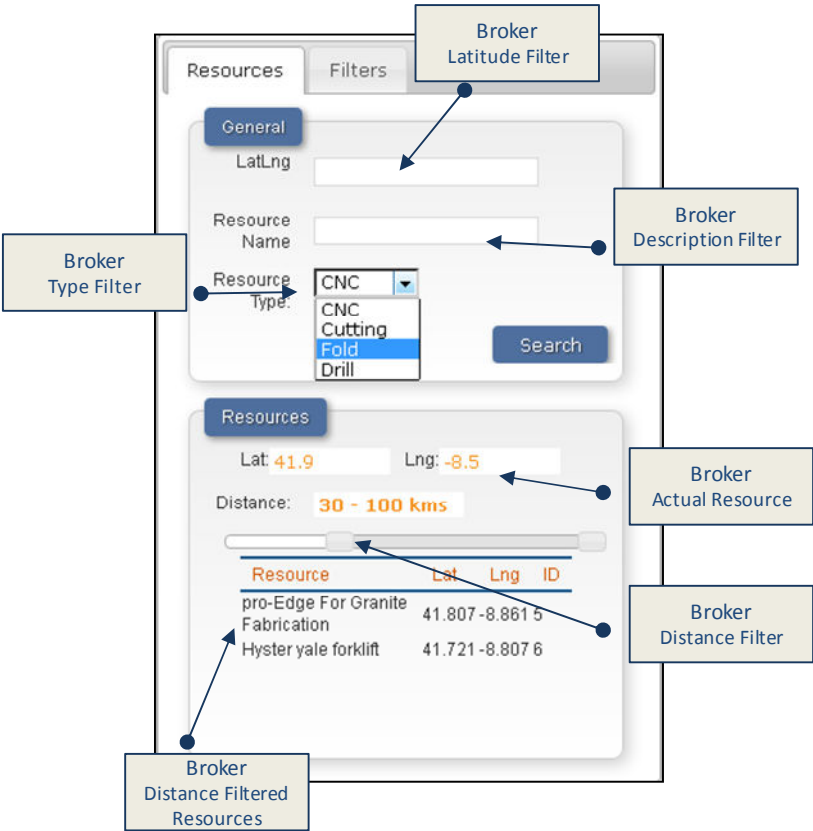


Figure 10.11 - Resources Selection (I)

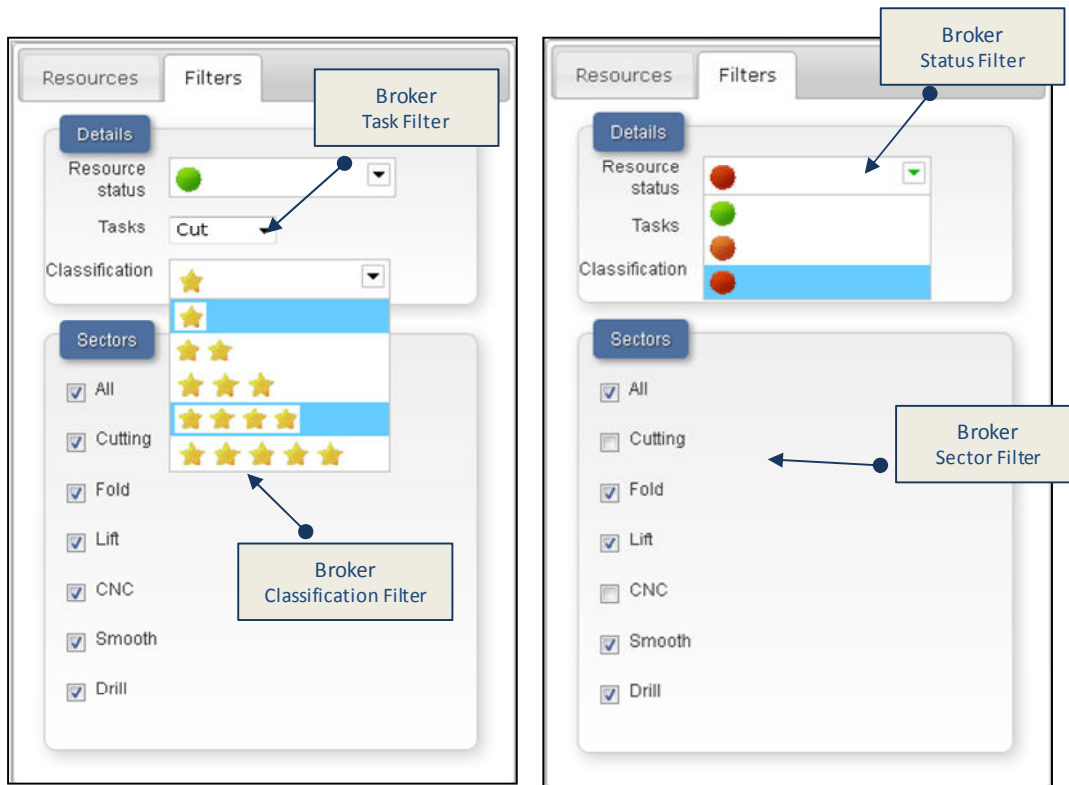


Figure 10.12 - Resources Selection (II)

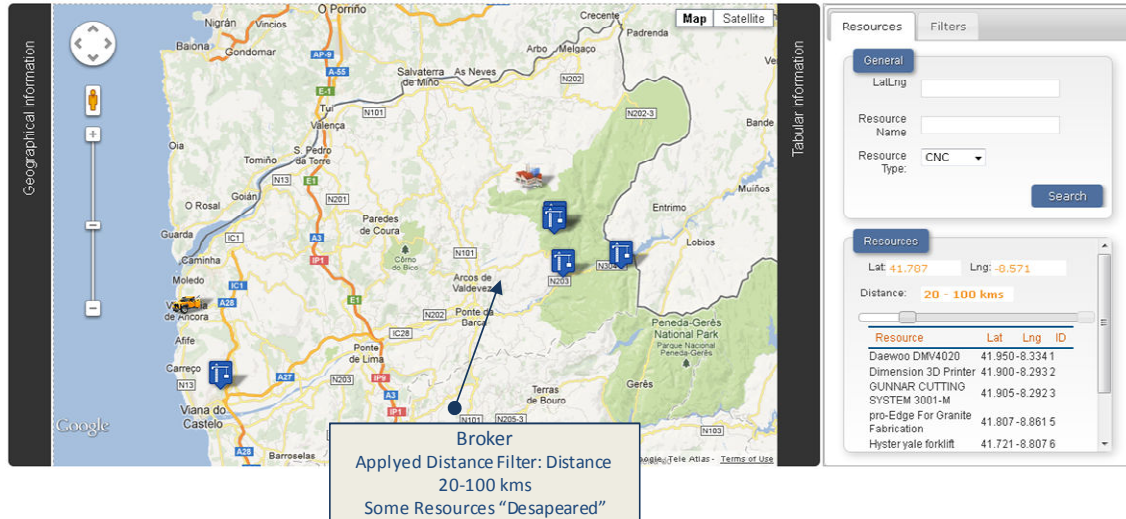


Figure 10.13 - Map representation of filtered resources

Over the map is also possible to analyze in detail the resource complementary information, including its status and classification and their communications channels (Figure 10.14)

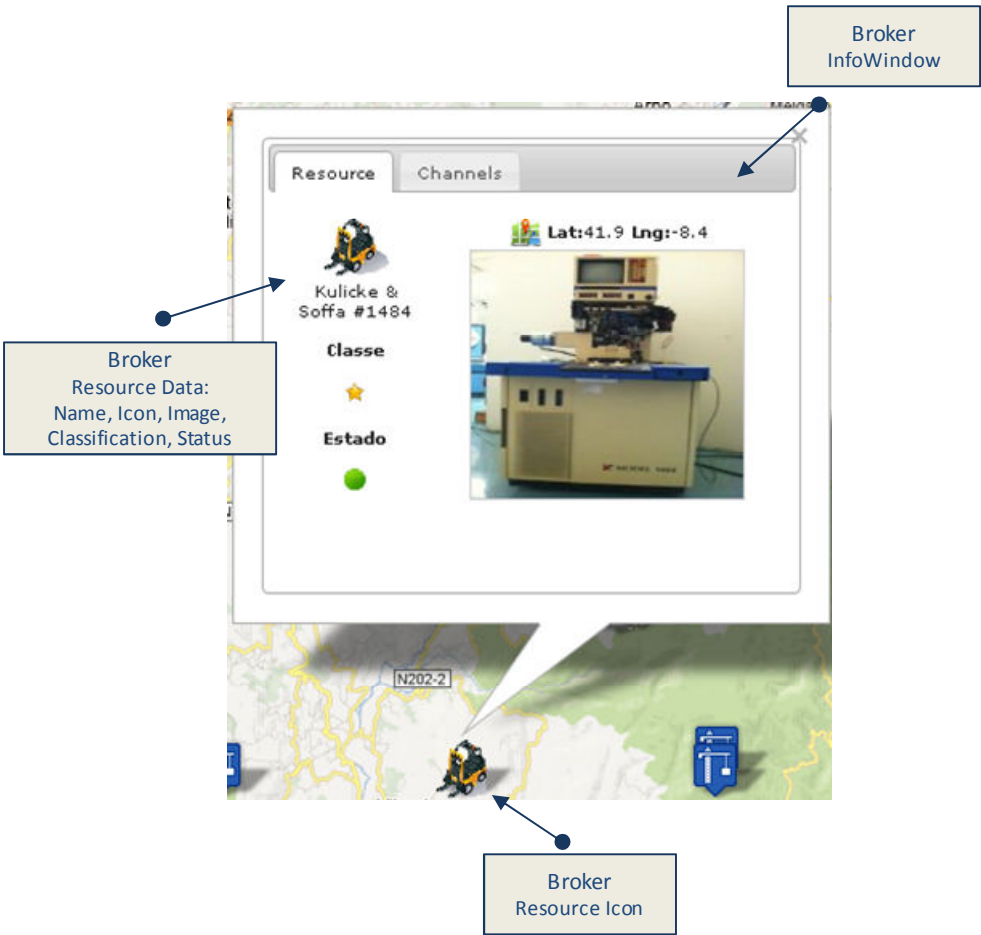


Figure 10.14 - Map InfoWindow with resource data

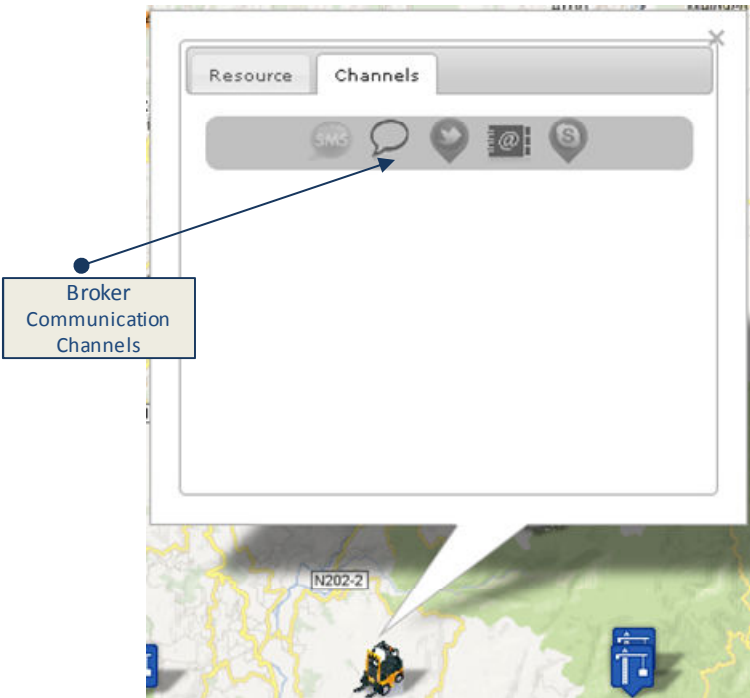


Figure 10.15 - Resource Communication Channels

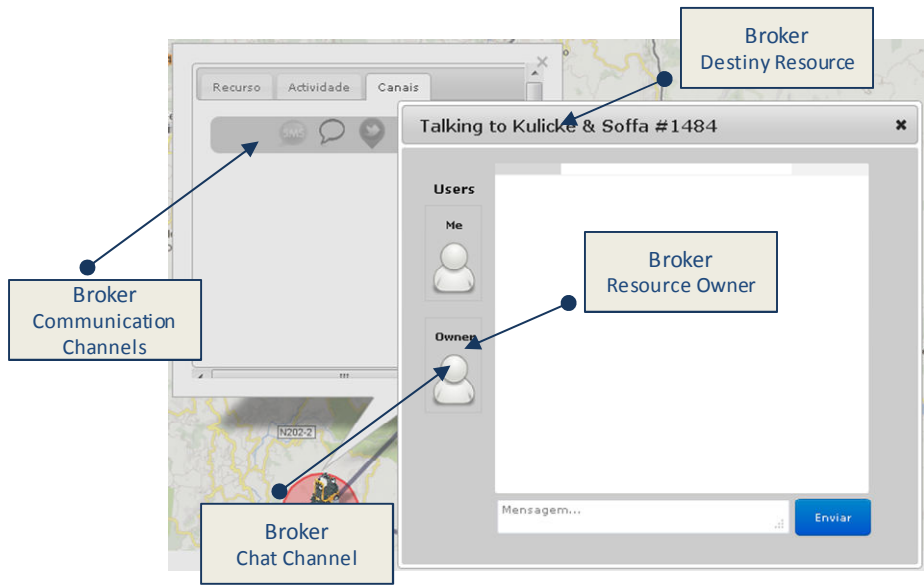


Figure 10.16 - Web Chat Channel

Reconfiguration Manager

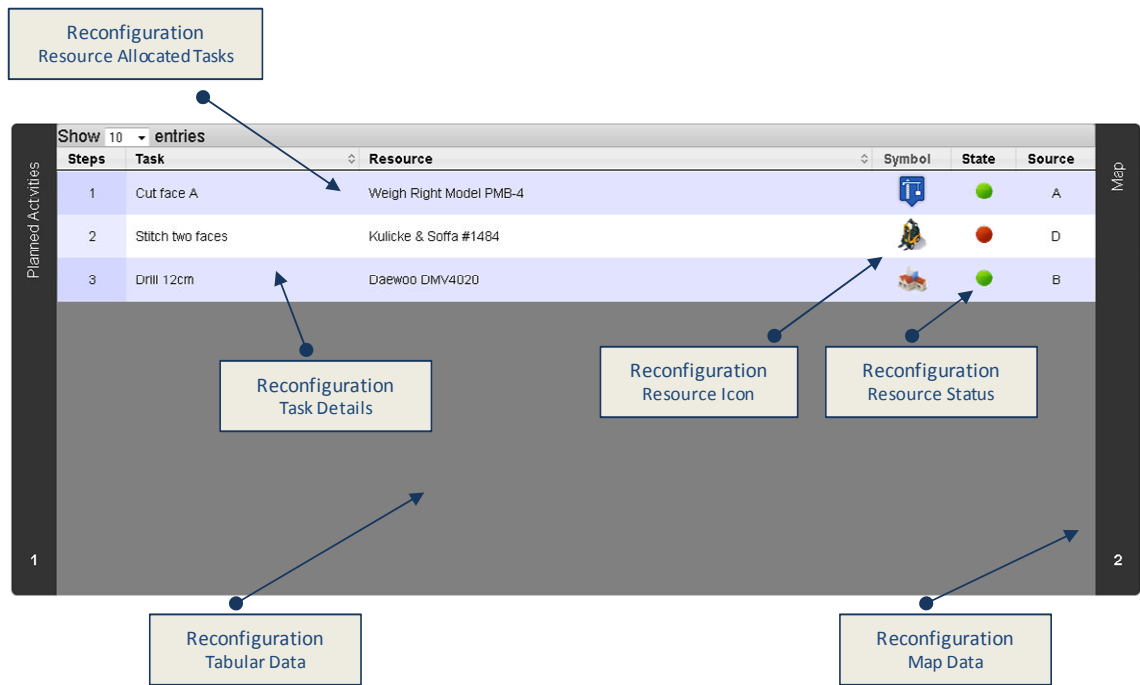


Figure 10.17 - Reconfiguration Manager

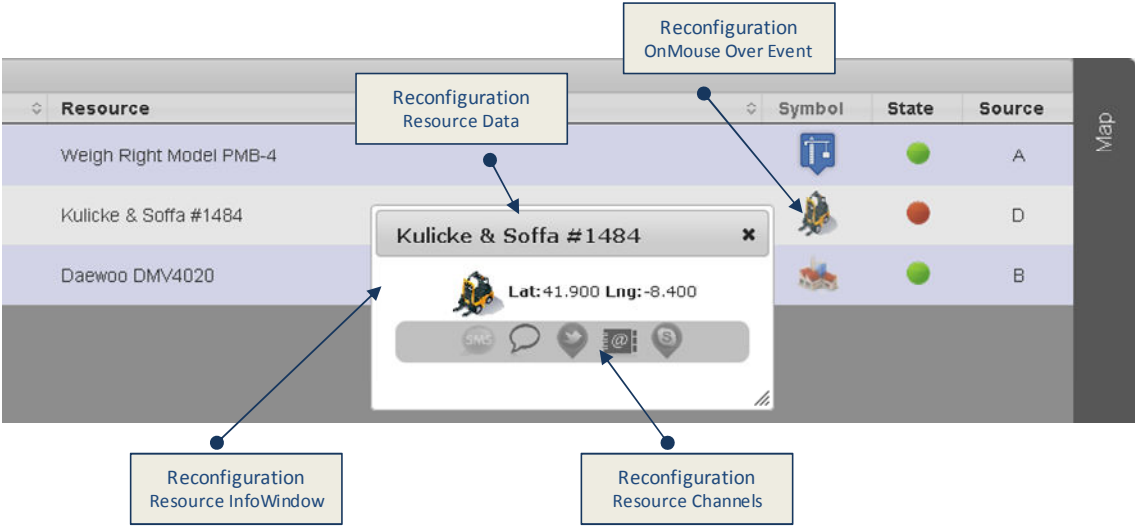


Figure 10.18 - Reconfiguration InfoWindows events

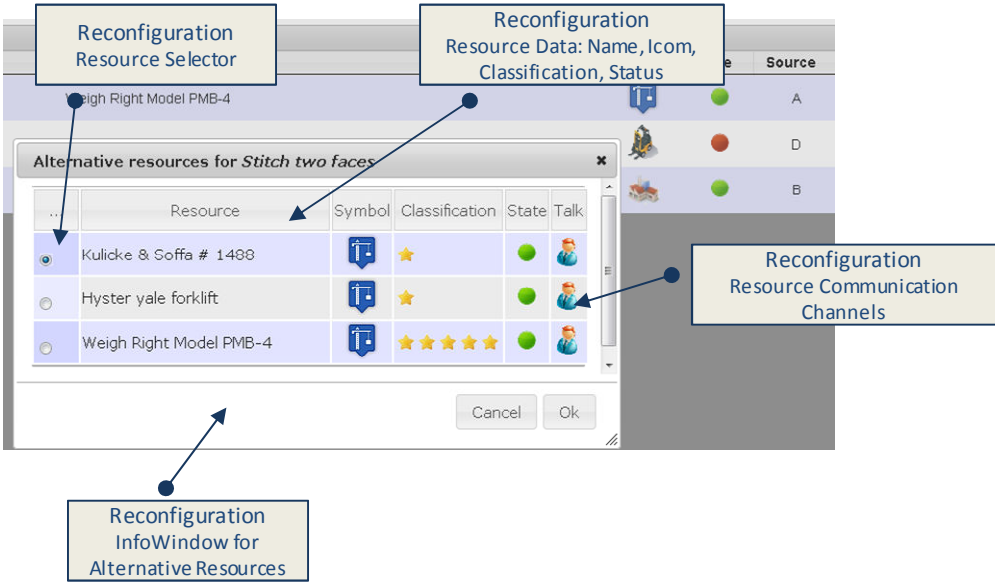


Figure 10.19 - Reconfiguration Alternative Resources Management

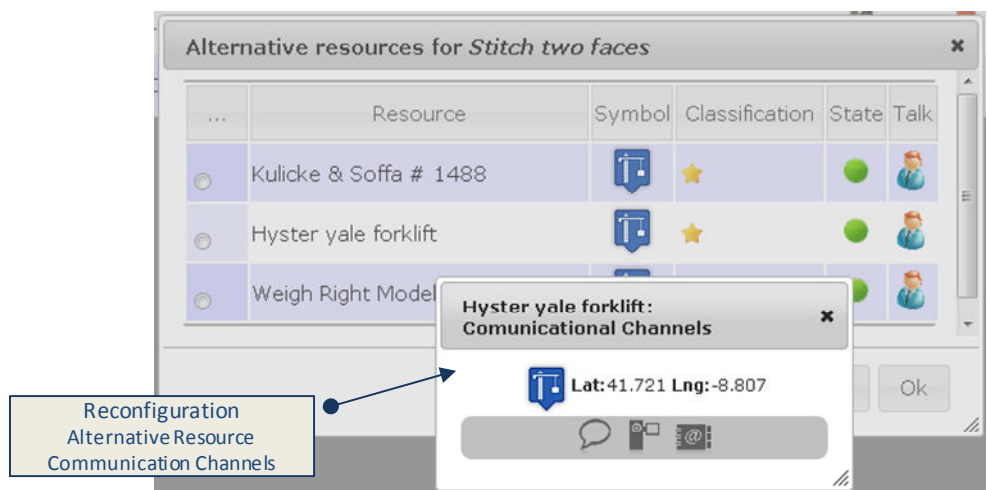


Figure 10.20 - Communication with Alternative Resource

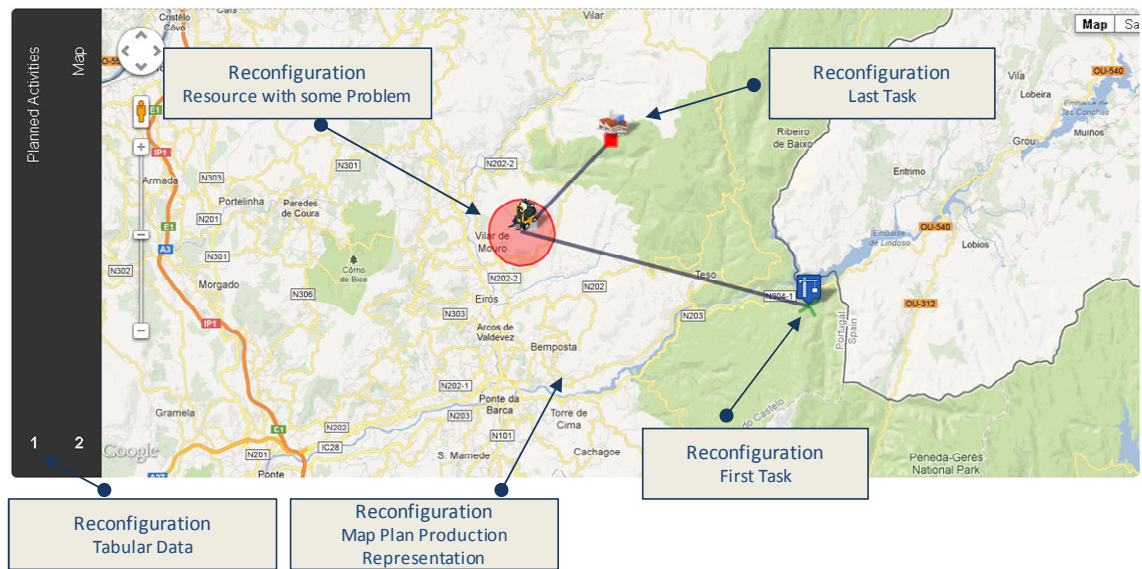


Figure 10.21 - Reconfiguration using the Map



Figure 10.22 - Reconfiguration Map Resource Event

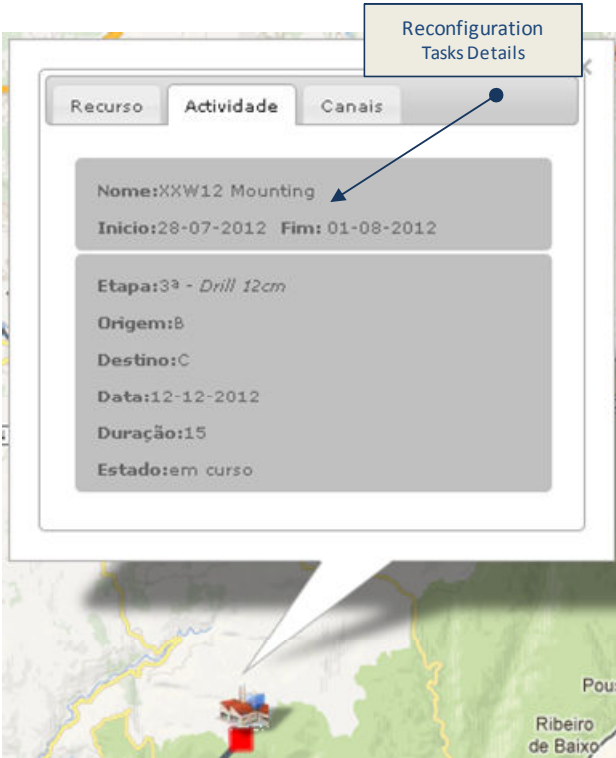


Figure 10.23 - Reconfiguration with Map Resource Details

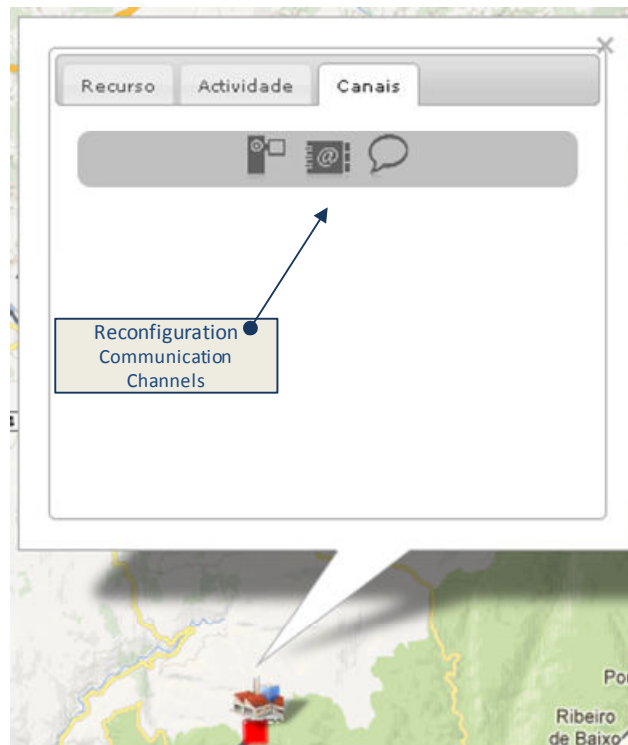


Figure 10.24 - Reconfiguration Alternative Resource Channel

11 APPENDIX B – MATERIAL FOR THE EXPERIMENTATION

Summary:

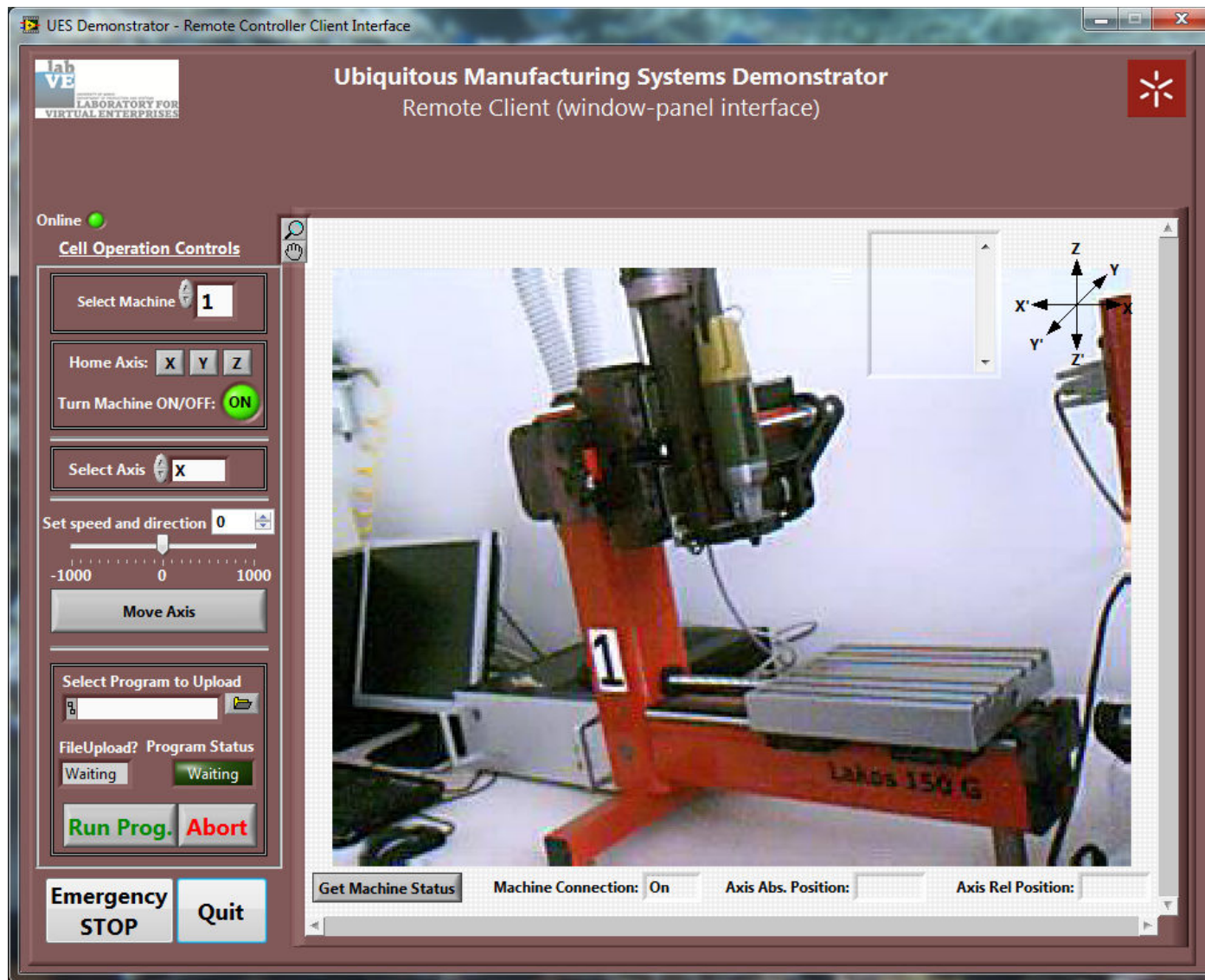
Part A		Part B		Part C	
Step 0	Step 1	Step 2	Step 3	Step 4	Step 5
(in class) 45 min		(in class) 30 min	(in class) 30 min	(in class) 60 min	(at home)

Regras

- No *Passo 1* o Autor define a sua ontologia no Portégé SW, e imprime o diagrama em PDF.
- No *Passo 2 – Interpretação de Ontologias*, cada autor interpreta a ontologia de outro. Utiliza a **Tabela A**, anotando na coluna 1 os termos sobre os quais tem uma opinião! Utiliza os valores da *escala de concordância* existente assim como a justificação para tal valor (na coluna 2)!
- No *Passo 3 – Revisão das Interpretações*, cada autor, revê a interpretação que foi feita à sua Ontologia inicial. Utilizando a coluna 3 da **Tabela A** manifesta a sua perspectiva utilizando a *escala de concordância* existente.
- No *Passo 4* – Ambos os autores conversam sobre as interpretações e revisões que fizeram. Ambos utilizam a coluna 4 da **Tabela A** para registar as respectivas conclusões.
- No *Passo 5* – Ambos os autores tentam chegar a uma nova ontologia que “satisfaça” a perspectiva de ambos.

Escala de concordância:

- ❶ - Concordo totalmente
- ❷ - Concordo
- ❸ - Discordo
- ❹ - Discordo completamente
- ❺ - Ignoro



Grupo:	Autor:	Ontologia	Passo 1

Grupo:	Autor da Ontologia:		1 - Concordo totalmente 2 - Concordo 3 - Discordo 4 - Discordo completamente 5 - Ignoro	
Taxonomia	PASSO 2		PASSO 3	
	Interpretação + Justificação 1 2 3 4 5		Revisão da Interpretação 1 2 3 4 5	
			PASSO 4	
			Análise Conjunta → Nova Ontologia	

Grupo:	Autores:	Nova Ontologia	Passo 5