UNIVERSITY OF NICE - SOPHIA ANTIPOLIS
DOCTORAL SCHOOL STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION
ET DE LA COMMUNICATION

PHD THESIS

to obtain the title of

PhD of Science

of the University of Nice - Sophia Antipolis
Specialty : Computer Science

Defended by

Olivier Commowick

Design and Use of Aatomical Atlases for Radiotherapy

Thesis Advisor: Grégoire Malandain

prepared at INRIA Sophia Antipolis, Asclepios Team

defended on February 14, 2007

Jury :

Reviewers :	Patrick Clarysse	-	CNRS (CREATIS)
	Louis Collins	-	McGill University
Advisor :	Grégoire Malandain	-	INRIA (Asclepios)
President :	Nicholas Ayache	-	INRIA (Asclepios)
Examinators :	Pierre-Yves Bondiau	-	Centre Antoine Lacassagne (Nice)
	Guido Gerig	-	University of North Carolina
	Vincent Grégoire	-	Université Catholique de Louvain
Invited :	Hanna Kafrouni	-	DOSISoft S.A.

Acknowledgments

Last thing to do :-)

Contents

CHAPTER 1

Introduction

Contents

1.1 Illustration Example

1.1.1 A subsection just for fun

Sorry I won't write your PhD here ;) This small text just to mention that this style supports writing with accents such as in french words (thèse, définir, ...). Also I put here a simple way to include an image. This is standard latex. For pdflatex compilation, the extension of the images is jpg. For latex compilation, this is ps or eps. The base folder containing images is set in formatAndDefs.tex, as well as the default extensions added to the image names.

1.2 An equation

Just to show argmin and partial derivative commands.

$$T = \arg\min_{T} E(T, R, F) \tag{1.1}$$

Regularization:

$$\frac{\partial T}{\partial t} = \Delta T \tag{1.2}$$

1.3 An other section

Showing a great bullet list environment:

- First point
- Second point

Figure 1.1: A nice image...

Chapter 2
No-name yet

2.1 Query taxonomy and re-usability formalization

This section presents the description and formalization of the different types of queries which *(i)* can be processed by our integration approach; and *(ii)* can be compared to previous integration requests in order to take advantage from previous integration plans. The query definition is introduced below.

Definition 1 *A query is defined as a n-tuple:*

$$Q := \langle A, R, S, C, w \rangle$$

where: A is a set of abstract services defining the query Q; R is a set of user preferences that can be defined over the data services or the entire query; S is a set of data services that were selected satisfying the restrictions defined by R to potentially rewrite the query Q; C is a set of compositions that were produced using the data services in S and satisfying the restrictions defined by R that potentially can answer the query Q; and w is the composition that were selected and executed to answer the query Q.

The query taxonomy proposed below is defined according to the type of relation that can be established between two queries. Queries are classified in four groups:

- *Group 1:* The data denoted by the answer of Q_1 is the same data expected by the answer of Q_2. For example, Q_1 and Q_2 retrieve patients that were infected by pneumonia.
- *Group 2:* The data denoted by the answer of Q_1 is a subset of the data denoted by the answer of Q_2. For example, Q_2 retrieves patients that were infected by pneumonia and Q_1 retrieves patients that were infected by pneumonia and treated by the doctor Lucas.
- *Group 3:* The data denoted by the answer of Q_1 is a superset of the data denoted by the answer of Q_2. For example, Q_2 retrieves patients that were infected by pneumonia and treated by the doctor Lucas, and Q_1 retrieves patients that were infected by pneumonia.
- *Group 4:* The data denoted by the answer of Q_1 is different of the data denoted by the answer of Q_2. For example, Q_2 retrieves patients that were infected by pneumonia and treated by the doctor Lucas, and Q_1 retrieves patients that were infected by pneumonia with admission in the hospital Edouard Herriot.

To understand the different types of query, basic concepts regarding *(i)* user requirements, *(ii)* requirements domain, *(iii)* requirements evaluation and *(iv)* comparable requirements should be introduced:

Definition 2 *An user requirement r is in the form $x \otimes c$, where x is an identifier; c is a constant; and $\otimes \in \{\geq, \leq, =, \neq, <, >\}$. The user requirement r could concern* *(i) the entire query, in this case noted as r_Q; or (ii) a single service, noted as r_S. For instance, the total response time is obtained by adding the response time of each service involved in the composition.*

Definition 3 *A requirement domain is a set of possible values which can be assumed by an user requirement r, represented by $Dom(r)$. For instance, a requirement domain "response time" includes the possible values associated to the response time user requirement. Each user requirement r_i has its own requirement domain D_i.*

Definition 4 *The evaluation of an user requirement r, indicated by $eval(r)$, returns a set of values $\{v_1, .., v_i\}$ that can be assigned to r such that $\{v_1, .., v_i\} \subset Dom(r)$.*

Definition 5 *Given two user requirements r_1 and r_2, both can be comparable, denoted by $r_1 \perp r_2$, if and only if: $Dom(r_1) = Dom(r_2)$.*

The thirteen types of queries included in the taxonomy described in the following sections are organized according to their groups.

2.1.1 Queries that can potentially be completely reusable

There are two types of queries belonging to this group. Given a previous query Q_1 stored in the query history and an incoming query Q_2, the types are: *(i)* Q_1 and Q_2 are completely equivalents (the simplest case); and *(ii)* Q_1 and Q_2 comprehend the same abstract services but Q_2 specifies user requirements less restrict than Q_1. The characteristics of these queries are described below:

a) *Query type 1:* the *first* type is the simplest case. The figure 2.1 illustrates the manner this query is represented. Given a previous query Q_1 and an incoming query Q_2, Q_1 is equivalent to Q_2 when: *(1)* both queries expect the same data as answer, which means they cover the same abstract services (Figure 2.1 - Data point of view). In this sense, the set of abstract service of Q_1, denoted as $Q_1.A$, is equals to the set of abstract services of Q_2, denoted as $Q_2.A$.

$$Q_1.A = Q_2.A$$

(2) For each user requirement r_i in $Q_1.R$, there is a user requirement r_j in $Q_2.R$ such that the evaluation of r_i is equal to the evaluation of r_j. Consequently, the score of $Q_1.R$ is equals to the score of $Q_2.R$. The *query type 1* and the equivalence between requirements are formally defined below.

Definition 6 *A set of user requirements R_1 is equivalent to a set of user requirements R_2, represented by $R_1 \equiv R_2$, if and only if: $\forall r_i \in R_1$, $\exists r_j \in R_2 \mid eval(r_i) = eval(r_j)$ and $|R_1| = |R_2|$.*

Definition 7 *Query Type 1 – a query Q_1 is equivalent to a query Q_2, if and only if: $Q_1.A = Q_2.A$ and $Q_1.R_1 \equiv Q_2.R_2$.*

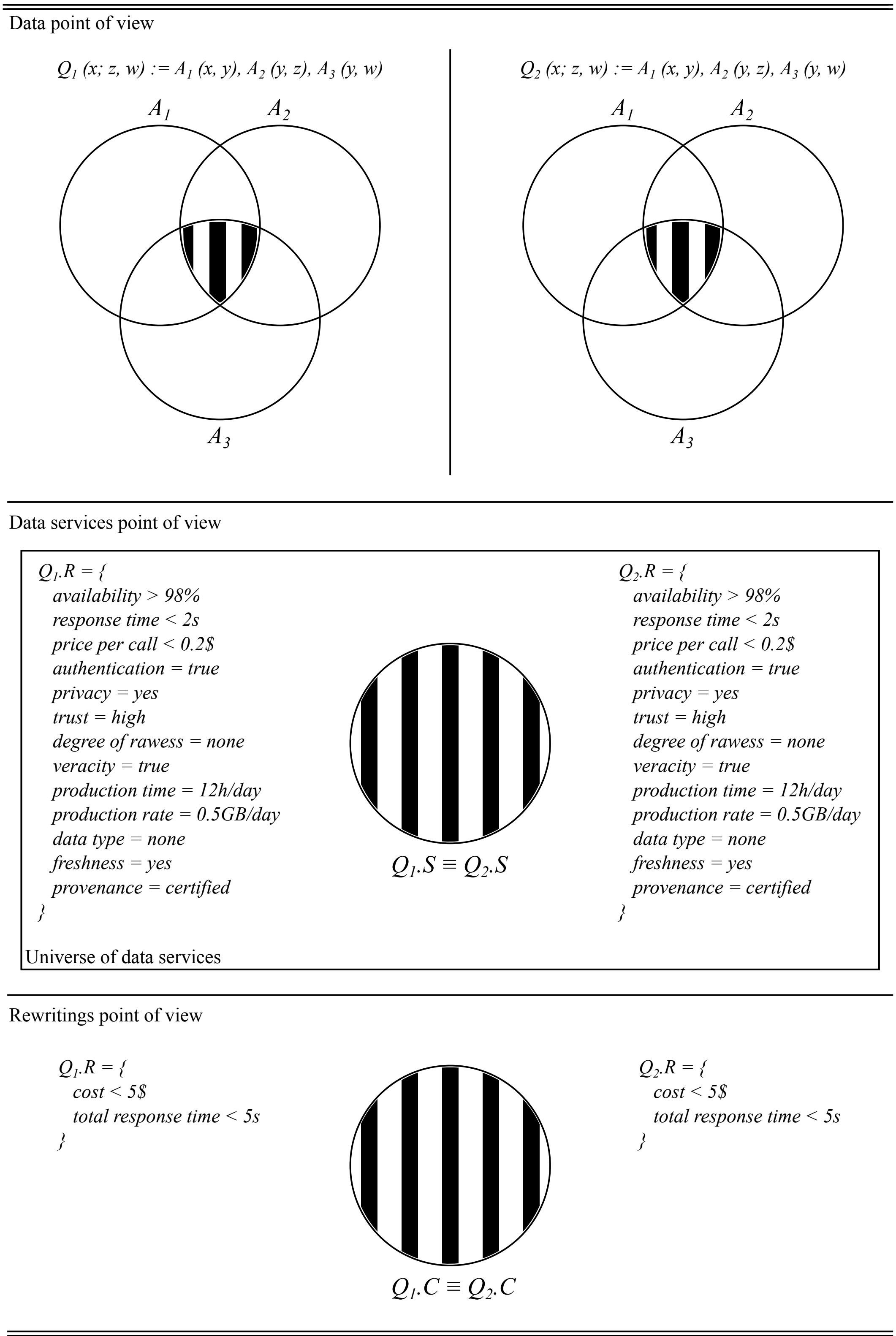

Figure 2.1: Query type 1 representation.

From the re-usability point of view, everything from Q_1 could be reused to answer Q_2. All data services filtered to the query Q_1, denoted $Q_1.S$, potentially could be reused in the query Q_2, excepting the ones that are not online in the exact moment. The set of compositions produced to the query Q_1, denoted as $Q_1.C$, potentially could be used to answer the query Q_2, excepting the ones using offline data services. Following, the reusability function for data services, rewritings and queries are presented.

The service reusability function, denoted as $reuse\ services(< query >, < query\ type >)$, receives as parameters a query and a query type, and returns a set of reusable data services.

Definition 8 *Given Q_1 and Q_2, queries of type 1, the reusability of services $reuse\ services(Q_1, type\ 1)$ selects and returns all data services in $Q_1.S$ online in the exact moment.*

$$Q_2.S = Q_2.S \ \cup \ \{ds_i\},\ \forall ds_i \in Q_1.S \mid ds_i \ is\ online.$$

The rewritings reusability function, denoted as $reuse\ rewritings(< query >, < query >)$, receives as parameters a previous query and the new query, and returns a set of reusable rewritings to answer the last one.

Definition 9 *Given Q_1 and Q_2, queries of type 1, the reusability of rewritings $reuse\ rewritings(Q_1,\ Q_2)$ selects and returns all rewritings in $Q_1.C$ which uses the services in $Q_2.S$.*

$$Q_2.C = Q_2.C \ \cup \ \{c_i\},\ \forall c_i \in Q_1.C \mid \{\ \forall ds_j \in c_i,\ \nexists ds_k \in c_i \mid ds_j \in Q_2.S\ and\ ds_k \notin Q_2.S\}.$$

b) Continuar aqui..query less restrict *Query type 1:* the *first* type is the simplest case. The figure 2.1 illustrates the manner this query is represented. Given a previous query Q_1 and an incoming query Q_2, Q_1 is equivalent to Q_2 when: *(1)* both queries expect the same data as answer, which means they cover the same abstract services (Figure 2.1 - Data point of view). In this sense, the set of abstract service of Q_1, denoted as $Q_1.A$, is equals to the set of abstract services of Q_2, denoted as $Q_2.A$.

$$Q_1.A = Q_2.A$$

(2) For each user requirement r_i in $Q_1.R$, there is a user requirement r_j in $Q_2.R$ such that the evaluation of r_i is equal to the evaluation of r_j. Consequently, the score of $Q_1.R$ is equals to the score of $Q_2.R$. The *query type 1* and the equivalence between requirements are formally defined below.

2.1.1.1 Query type 2: Q_2 is a subset of Q_1

The *second* type deals with *query subsets* due to more restrict user requirements. Given two queries Q_1 and Q_2, Q_2 is a subset of Q_1 when:

a) They expect the same data as answer, which means they cover the same abstract services. For instance, the set of abstract service of Q_1, denoted as $Q_1.A$, is equals to the set of abstract services of Q_2, denoted as $Q_2.A$.

$$Q_1.A = Q_2.A$$

b) For all user requirement r_i in $Q_2.R$, there is at least one r_j in $Q_1.R$ such that the evaluation of r_i is contained in the evaluation of r_j. For all r_k in $Q_2.R$, there is no r_l in $Q_1.R$ such that the evaluation of r_l is contained in the evaluation of r_k. Consequently, the score of $Q_1.R$ is lower than the score of $Q_2.R$. The definition of more restrict requirements is presented below.

Definition 10 *Given a set of user requirements R_1 and R_2, R_1 is more restrict than R_2, represented by $R_1 \ \triangleright\ R_2$, if and only if: $\forall r_i \in R_1$, $\exists r_j \in R_2$, $\nexists r_k \in R_2 \mid eval(r_i) \subset eval(r_j)$ and $eval(r_k) \subset eval(r_i)$ and $|R_1| = |R_2|$.*

From the re-usability point of view, a subset of the data services filtered to the query Q_1 which are *online* in the moment, $online(Q_1.S)$, could be reused in the query Q_2. This fact occurs due to the more restrict requirements imposed by Q_2. With respect to the compositions, a subset of the rewritings produced to the query Q_1 could also be used to answer the query Q_2. These rewritings should use the data services in $online(Q_1.S)$, denoted as $available(Q_1.C)$, and respect the more restrict requirements defined in Q_2. The query type 2 definition is presented below.

Definition 11 *Query Type 2 – a query Q_1 is a subset of a query Q_2, if and only if: $Q_1.A = Q_2.A$ and $Q_1.R_1 \triangleright Q_2.R_2$*

Appendix A

Appendix Example

A.1 Appendix Example section

And I cite myself to show by bibtex style file (two authors) [Commowick 2007].

This for other bibtex stye file : only one author [Oakes 1999] and many authors [Guimond 2000].

Bibliography

[Commowick 2007] Olivier Commowick and Grégoire Malandain. *Efficient Selection of the Most Similar Image in a Database for Critical Structures Segmentation.* In Proceedings of the 10th Int. Conf. on Medical Image Computing and Computer-Assisted Intervention - MICCAI 2007, Part II, volume 4792 of *LNCS*, pages 203–210. Springer Verlag, 2007. (Cited on page 9.)

[Guimond 2000] A. Guimond, J. Meunier and J.-P. Thirion. *Average Brain Models: A Convergence Study.* Computer Vision and Image Understanding, vol. 77, no. 2, pages 192–210, 2000. (Cited on page 9.)

[Oakes 1999] David Oakes. *Direct Calculation of the Information Matrix via the EM Algorithm.* J. R. Statistical Society, vol. 61, no. 2, pages 479–482, 1999. (Cited on page 9.)

Design and Use of Numerical Anatomical Atlases for Radiotherapy

Abstract: The main objective of this thesis is to provide radio-oncology specialists with automatic tools for delineating organs at risk of a patient undergoing a radiotherapy treatment of cerebral or head and neck tumors.
To achieve this goal, we use an anatomical atlas, i.e. a representative anatomy associated to a clinical image representing it. The registration of this atlas allows to segment automatically the patient structures and to accelerate this process. Contributions in this method are presented on three axes.
First, we want to obtain a registration method which is as independent as possible w.r.t. the setting of its parameters. This setting, done by the clinician, indeed needs to be minimal while guaranteeing a robust result. We therefore propose registration methods allowing to better control the obtained transformation, using outlier rejection techniques or locally affine transformations.
The second axis is dedicated to the consideration of structures associated with the presence of the tumor. These structures, not present in the atlas, indeed lead to local errors in the atlas-based segmentation. We therefore propose methods to delineate these structures and take them into account in the registration.
Finally, we present the construction of an anatomical atlas of the head and neck region and its evaluation on a database of patients. We show in this part the feasibility of the use of an atlas for this region, as well as a simple method to evaluate the registration methods used to build an atlas.
All this research work has been implemented in a commercial software (Imago from DOSIsoft), allowing us to validate our results in clinical conditions.
Keywords: Atlas-based Segmentation, non rigid registration, radiotherapy, atlas creation