MSc in Computer Science
Module: MCOM 0177 Computer Science MSc Project

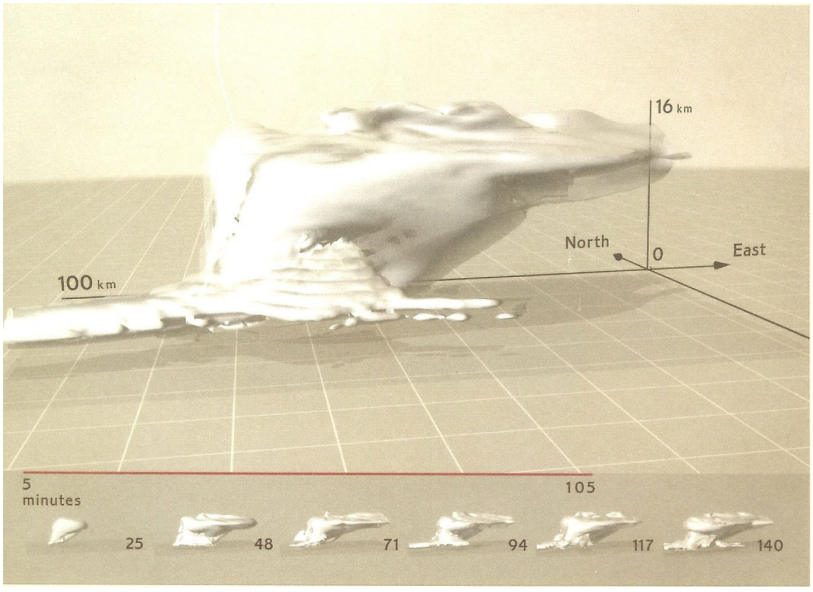
Design and development of a prototype addressing spatio-temporal environmental vector data management, analysis and delivery using Open Source technology

General framework and case study focused on groundwater management in a coastal area

Ezio Crestaz / SRN 08174905
Viale Guarnieri, 2 - 61032 Fano (PU), Italy
Phone: (0039) 0721 862724
Email: ezio.crestaz@giscience.it

Supervisors:
Dr. Vito Veneziano
Dr. Ian Bradford

Total words (excluded footnotes, references, appendices)**: 9770**



Fano, January 31st 2011

Cover image

3D spatio-temporal visualization of a numerical simulation of a severe storm

Redesigned by Tufte, 1997 based on original visualisation by Arrot M., Bajuk M., Bushell C.B., Thingvold J. and Yost J.B. of the National Center for Supercomputing Applications, Un. of Illinois at Urbana Campaign

TABLE OF REFERENCES

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AJAX	**A**syncronous **J**avascript **A**nd **X**ML
ANSI	**A**merican **N**ational **S**tandards **I**nstitute
BLOB	**B**inary **L**ong **OB**ject
BP	**B**ritish **P**etroleum Inc.
BSD	**B**erkley **S**tandard **D**istribution license
CTE	**C**ommon **T**able **E**xpression
DBMS	**D**ata**B**ase **M**anagement **S**ystem
DCL	**D**ata **C**ontrol **L**anguage
DDL	**D**ata **D**efinition **L**anguage
DEM	**D**igital **E**levation **M**odel
DML	**D**ata **M**anipulation **L**anguage
DOM	**D**ocument **O**bject **M**odel
DTM	**D**igital **T**errain **M**odel
EAR	**E**ntity **A**ttribute **R**elationship
EER	**E**xtended **E**ntity **R**elationship
EMS	**E**nvironmental **M**anagement **S**ystem
EPSG	**E**uropean **P**etroleum **S**urvey **G**roup
ESDA	**E**xploratory **S**patial **D**ata **A**nalysis
ESRI	**E**nvironmental **S**ystems **R**esearch **I**nstitute
ESTAT	**E**xploratory **S**patio-**T**emporal **A**nalysis **T**oolkit
EU	**E**uropean **U**nion
FD	**F**inite **D**ifference
FE	**F**inite **E**lement
FGDC	**F**ederal **G**eographic **D**ata **C**ommittee
FK	**F**oreign **K**ey
FSF	**F**ree **S**oftware **F**oundation
FOSS	**F**ree and **O**pen **S**ource **S**oftware
FOSS4G	**F**ree and **O**pen **S**ource **S**oftware for **G**eomatics
GI	**G**eographic **I**nformation
GIS	**G**eographical **I**nformation **S**ystem
GISc	**G**eographical **I**nformation **Sc**ience
GiST	**G**enerali**z**ed **S**earch **T**ree
GML	**G**eographic **M**arkup **L**anguage
GPL	**G**eneral **P**ublic **L**icense
GRASS	**G**eographic **R**esources **A**nalysis **S**upport **S**ystem
GUI	**G**raphical **U**ser **I**nterface
IGWMC	**I**nternational **G**round **W**ater **M**odeling **C**enter
ISO	**I**nternational **S**tandard **O**rganization

ISO TC/211	**I**nternational **S**tandard **O**rganization **T**echnical **C**ommittee **211**
IT	**I**nformation **T**echnology
JRC	**J**oint **R**esearch **C**enter
JTS	**J**ava **T**opology **S**uite
LBS	**L**ocation **B**ased **S**ervice
LGPL	**L**esser **G**eneral **P**ublic **L**icense
LiDAR	**Li**ght **D**etection **A**nd **R**anging
M-M	**M**any to **M**any
MNDNR	**MiN**nesota **D**epartment of **N**atural **R**esources
NASA	**N**ational **A**merican **S**pace **A**gency
NGO	**N**on **G**overnmental **O**rganization
OGC	**O**pen **G**IS **C**onsortium
OO	**O**bject **O**riented
OR	**O**bject **R**elational
ORDBMS	**O**bject **R**elational **D**ata**B**ase **M**anagement **S**ystem
OS	**O**pen **S**ource
PCP	**P**arallel **C**oordinate **P**lot
PK	**P**rimary **K**ey
PPGIS	**P**ublic **P**articipation **G**eographical **I**nformation **S**ystems
RCP	**R**ich **C**lient **P**latform
RDBMS	**R**elational **D**ata**B**ase **M**anagement **S**ystem
SDE	**S**patial **D**ata **E**ngine
SDI	**S**patial **D**ata **I**nfrastructure
SFS	**S**imple **F**eature interface **S**tandard
SQL	**S**tandard **Q**uery **L**anguage
SQL/MM	**S**tandard **Q**uery **L**anguage/**M**ulti **M**edia
SRID	**S**patial **R**eference **ID**
UML	**U**nified **M**odelling **L**anguage
UMN	**U**niversity of **MiN**nesota
URL	**U**niform **R**esource **L**ocator
VGI	**V**olunteered **G**eographic **I**nformation
ViSC	**Vi**sualization for **S**cientific **C**omputing
WFS	**W**eb **F**eature **S**ervice
WFS-T	**T**ransactional **W**eb **F**eature **S**ervice
WKB	**W**ell **K**nown **B**inary
WKT	**W**ell **K**nown **T**ext
WMS	**W**eb **M**apping **S**ervice
XML	e**X**change **M**arkup **L**anguage
1-M	**1** to **M**any

ACKNOWLEDGMENTS

My academic and professional background in GIScience and groundwater modelling assisted me in preliminary assessment of key issues in environmental spatio-temporal data management, analysis and delivery. On the other hand, I had to face my traditional bias towards proprietary solutions, particularly ESRI ArcGIS. I had to spend quite a lot of time reviewing and investigating major technical and organizational benefits/bottlenecks of open source and free solutions, as well as to design and develop spatial database and web mapping application prototypes.

Many people helped me as well along this complex path.
Particularly I must acknowledge Dr. Maurizio Gibin, lecturer in GISc at Birkbeck College, for the in-depth discussions on most of the dissertation topics, Dr. Marco Foi, GIS technician at the Un. of Milano, who provided both hosting and administration support in setting up a complete test open source stack and useful web programming tips, and Dr. Joana Barros, director of the MSc in GISc at Birkbeck College, who called me to teach advanced modules on database design and environmental GIS in London, providing the opportunity to investigate current issues from different perspectives.

Although the support from my tutors is an obvious part of their duties, I can not omit a warm acknowledgement to Dr. Vito Veneziano for being proactive in pointing me at open source issues and very supportive, and to his family for their warm friendly welcome when I travelled to London.

I must acknowledge the support of my family, Anna, Pietro Noah and Noel as well, particularly for all the times I have been nervous at home for the noise, while spending most of the time on my own to honour the MSc deadlines.
And finally a thought to my parents, to Rita and Sergio.

"*Any GIS project which adopts a pure technology focus is doomed to failure and customisation projects are no exception*"

Maguire, 2005

INTRODUCTION

Every day, huge amount of environmental, geo-referenced data are collected worldwide by means of regional and local monitoring networks (Maasdam, 2000), integrated with remote sensed data sources, and made available at different spatial scale and temporal frequency (i.e. Landsat and LiDAR[1] ; Lillesand and Kiefer, 2000) to a variety of users and audiences.

These data play a key role for detailed environmental status assessment, focusing on such different components as surface water, groundwater, soil, forests, air, addressing specific risks, generally of relevance to socio-economic development (i.e. groundwater in agriculture; Dosi, 2001; UNEP, 2011) as well human life, such as flooding, forest fires, supply shortage, contamination, salt water intrusion, just to recall few of them. Recent BP oil spill in Gulf of Mexico (Fig. 1) and the death of Aral sea[2] (**Fig. 2**), following wicked Amudarja river management policies, are examples of large scale transboundary environmental emergencies/disasters.

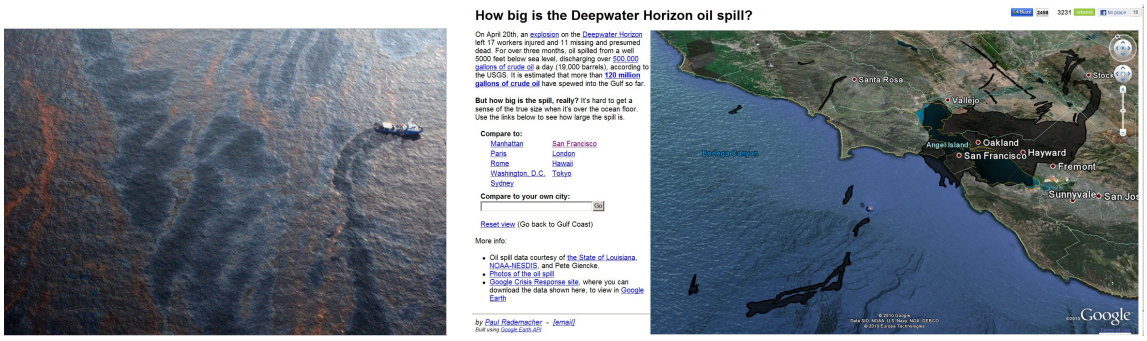
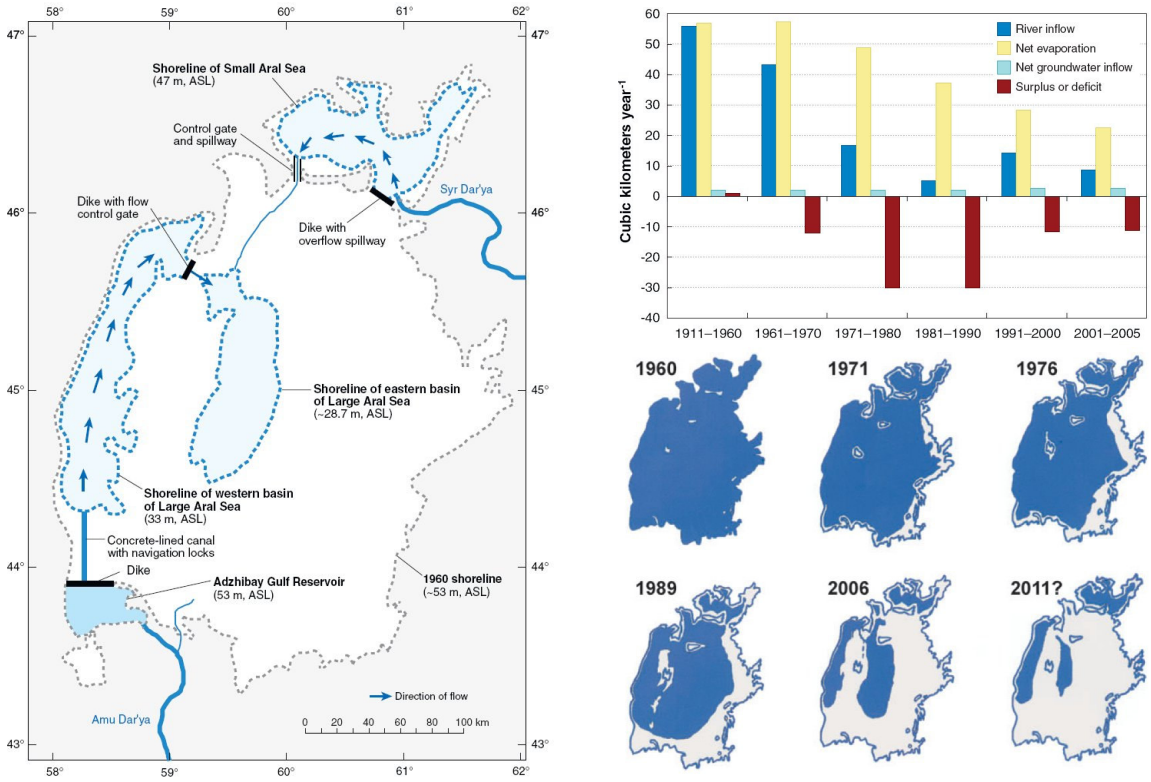


Fig. 1 - BP oil spill in Gulf Mexico and Google Maps mashup showing its areal extension against the San Francisco Bay area

[1] http://glovis.usgs.gov/ (USGS, 2010) http://www.lidarmap.org/

[2] http://www.unesco.org/water/news/newsletter/232.shtml#know



Micklin, 2007

Fig. 2 - Aral sea: detailed shores map, average annual water balance and shrinking boundaries

As "*silos of data*"[3] have been growing with time, particularly in last decades due to major advancements in remote data collection, GIS[4] theoretical framework (Worboys, 1995; 2005) and technology have been evolving, since late 50ies, to effectively address, among others, environmental issues and management policies. Since the first commercial GIS platform in the 70ies, ESRI[5] (2011) ArcInfo, a true revolution has taken place, including the migration towards client-server architectures, the wide diffusion of spatial databases (MacDonald, 1999; Zeiler, 1999; Rigaux et al., 2002) and geoweb services (Lake et al., 2004; Peng and Tsou, 2003; Turton, 2010) and, generally speaking, a relevant effort towards standardisation (ISO, 2010;

[3] The concept has been introduced by former USA vice-president Al Gore, in his famous speech on digital earth, referring to satellite images. http://en.wikipedia.org/wiki/Digital_Earth

[4] According to Burrough and McDonnell (1986) definition, a "*Set of tools for collecting, storing, retrieving at will, transforming and displaying spatial data from the real world for a particular set of purposes*"

[5] Not surprisingly, ESRI stays for **E**nvironmental **S**ystems **R**esearch **I**nstitute, www.esri.com

Kresse and Fadaie, 2004; OGC, 2010), a complex ongoing process as humoristically captured by Dilbert (Fig. 3).

Dilbert and Web Services by Scott Adams
http://www.dilbert.com/

Fig. 3 - At what extent are standards developed?

While many other commercial players have been entering into the market, including major database giants (i.e. Oracle[6], 2010), the Open Source (OS) revolution has been taking place as well (Raymond, 2000), leading to many relevant projects in spatial databases (i.e. PostgreSql/PostGIS; Ramsey, 2007; Obe and Hsu, 2010; Avén, 2010), desktop GIS (i.e. GRASS, uDig, QuantumGIS, OpenJump, gvSIG), geographic servers (Map Server, Kropla, 2005; Geoserver, 2010) and geoweb services (EPA, 2009), geo-enabling web applications with rich contents and functionalities, towards a higher level of integration and user interaction experience (i.e, Google Maps, Bing, 2010; Purvis et al., 2006; Svennerberg, 2010; OpenLayers, 2010; Google Chart Motion[7]). Refer to existing literature for an overview of OS software (Dunfey et al., 2006; Neteler and Raghavan, 2006; Sanz-Salinas and Montesinos-Lajara, 2009; Steiniger and Bocher, 2009), and specifically on its impact in water sector (Daoyi, 2008).

Unfortunately, most of the GIS original tradition has been focusing on a file-based paradigm (i.e. the *de-facto* shape standard syndrome; ESRI, 1998); basically due to hardware and relational database platforms limitations, loose coupling solutions have been for long time the only viable option, leading to separate management of geographic information and alphanumeric attributes through the adoption of so-called dual architectures[8].

[6] Quite interestingly, also Oracle XE (Bobrowski, 2006), the entry level small footprint database based on Oracle database 10g technolgy, supports Locator, a subset of Spatial Oracle extension features http://www.oracle.com/technology/products/database/xe/index.html
Main limitations compared with priced options include data storage (maximum up to 4 Gb) and systems performance (1 Gb memory and 1 processor only, on Server side).

[7] http://code.google.com/intl/it/apis/visualization/documentation/gallery/motionchart.html
http://www.gapminder.org/

[8] A typical dual architecture provides specialized binary storage for geographic data, due to their size and complexity (i.e. multipolygon richness, topology, spatial indexes for effective access as RTree family), clearly distinct from related alphanumeric attributes possibly stored to a relational database

More recently, emerging Object Relational (OR) model (for OO concepts: Ambler, 2004; NA; Britton and Doake, 2005) and more powerful hardware supported full integration of geographical information in database, further to migration of basic as well as advanced geocomputation features (i.e. spatial operators, geocoding, network analysis) from traditional GIS to database domains. Despite at quite different levels of support and standardisation, GIS systems have generally been integrated with databases through direct connectors or some middleware (i.e. ArcSDE; ESRI, 1999), although generally providing poor querying capabilities[9] or taking undesired control over data storage through proprietary formats[10]. Also mostly spatial-analysis oriented systems (i.e. Idrisi[11]) or advanced environmental analysis systems (i.e. FEFLOW for groundwater modelling in DHI-WASY, 2010; theory in Bear and Verruijt, 1986) have been adding basic database support in most recent releases.

Provided that such trends, favoured by relevant advancements in database theory and platforms (Connolly and Begg, 2010; pp. 921-970), has lead to a better integration of databases with GIS, simulation modelling (Anderson and Woessner, 2002) and advanced visualisation for ESDA[12] (Tufte, 1997; 2001; Anselin et al., 2007), still traditional differences between the two domains persist[13], leading – among others – to such problems as:

- adoption of non-standard proprietary systems, often suffering for commercial policies, which strongly limit consistent implementations towards effective data interoperability;

[9] Among referred OS Desktop platforms, OpenJump results to be the only one permitting to issue a complex geographical query to underlying spatial database, geographically visualize its output and possibly save it to a new spatial table. All other platforms, including ArcGIS, remain limited to elementary selections.

[10] ArcGIS provides an interesting example of such an issue, storing geographic data to a BLOB shape field or SDE geometry. For completeness, standard geographic types as Oracle SDO_GEOMETRY (through proper ArcSDE setting) and/or direct connection are also supported.

[11] http://www.clarklabs.org/

[12] GeoDa Center for Geospatial Analysis and Computation, Arizona State Un., USA
http://geodacenter.asu.edu/
The work of Edward Tufte at http://www.edwardtufte.com/tufte/
See also Kraak, 2005

[13] These differences are partly due to contrasting cultures, database stemming from IT and GIS from applied geography, difference which is partly overcome by GISc (ref. the Big Book of GIS, Longley et al., 2005[a])

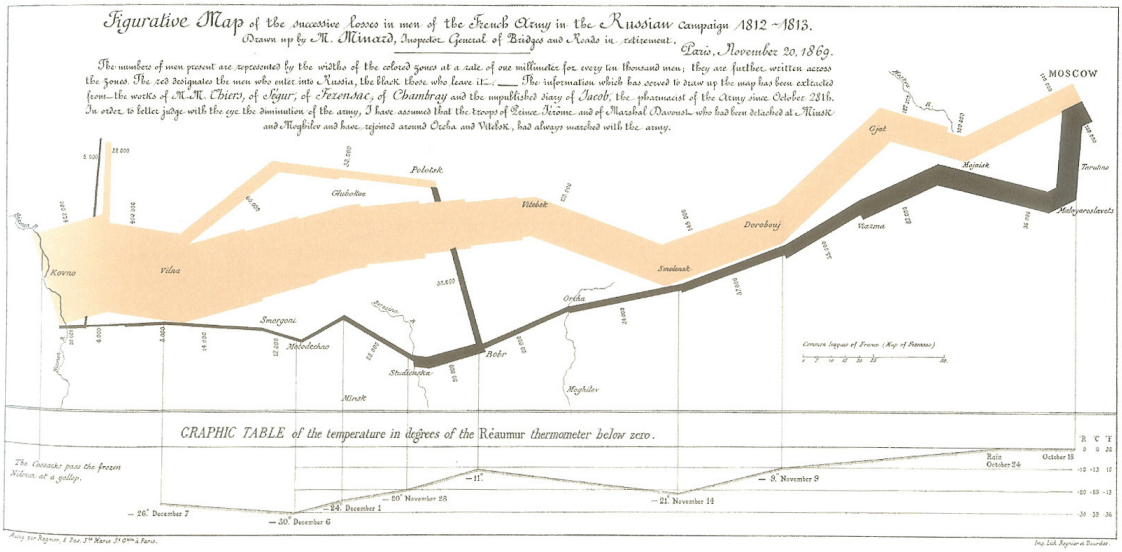
- prevalence of loose coupling strategies (Weaver et al., 2003) for integration of different state-of-the art components, namely storage, GIS analysis, modelling and visualisation, often leading to functionalities duplication and poor integration;

- limited or controversial awareness about key value of GI and proper data management policies; despite the great efforts devoted to national and continental SDI design and development[14] (Masser, 2005), most projects, particularly when based on short budgets, still tend to adopt ad-hoc approaches.

Coming to main dissertation focus, much effort has been spent in last decades to research in and applications development for effective integration of spatio-temporal dimensions[15] (Langran, 1992; Worboys, 1998; Hogeweg, 2000; Ott T. and Swiaczny F., 2001; Kothuri et al., 2004), a traditional challenge as classic work of Charles Joseph Minard highlights (Fig. 4). Further to advanced spatial database models, also Desktop GIS and web mapping tools have been coping with such additional challenges (van Deursen, 1995) as time and (relevant to most environmental applications) 3D, by addressing graphical and statistical analysis of spatio-temporal patterns (Forer, 1998). Today, these advancements more effectively support user in submitting complex queries, while addressing complex modelling and environmental problems, as contaminated plumes transport and fate.

[14] INSPIRE GeoPortal: http://www.inspire-geoportal.eu/

[15] http://www.esri.com/software/arcgis/geodatabase/index.html
http://resources.arcgis.com/content/data-models



Minard in Marey, 1885 – Translated version reported in Tufte, 2001; p. 41

Fig. 4 - Space, time and attributes of Napolean Russian campaign

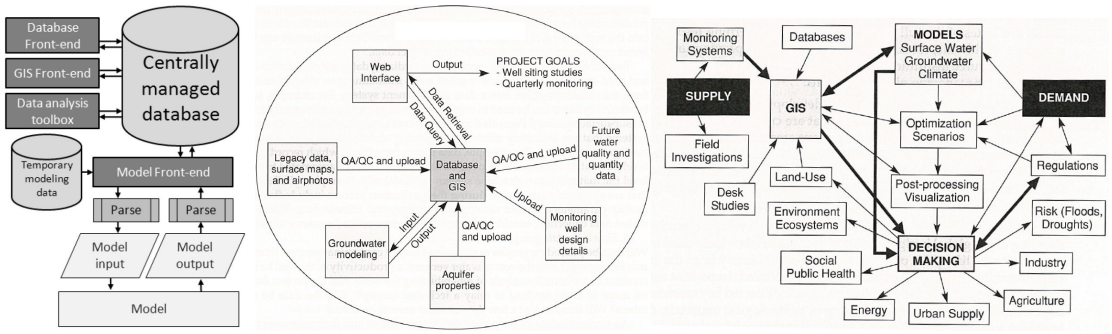
Definitely, dissertation investigates the key challenges of a generic environmental spatio-temporal database management framework, based on advancements in OS community and particularly the PostGIS (2010) extension for PostgreSQL (2010). The issues of effective access and visualisation of spatio-temporal data (Frank, 1998; Hazelton, 1998), GIS desktop customization (Maguire, 2005[16]; spatial Java programming in Wood, 2002) and the role of geographic servers are then briefly summarized, before focusing on the design and development of a front-end web mashup application based on Google Maps API v. 3 (Brown, 2006; Chow, 2008; Jankowski et al., 2007; Peterson, n.a.; Svennerberg G., 2010;).

Based on the spatio-temporal software prototype, a schematic test case study addressing groundwater monitoring data collection has been set up, in order to assess overall system effectiveness. The case study, with more than 1000 point locations and around 37000 monitored data, can be considered consistent with the requirements of typical few years ground (not automated) environmental monitoring projects, at local scale, as in the framework of

[16] Much effort is devoted to customization, also for GIS commercial platforms. See Maguire (2005) for estimates of overall customization costs (up to 30% vs. total costs, including licenses and personnel) and the literature on VBA-ArcObjects customization of ArcGIS platform to appreciate the complexity of these frameworks (Burke, 2003; Chang, 2005; Zeiler; 2001[a], 2001[b]). For temporal visualization in ArcGIS, refer to DHI-WASYTemporal Analyst extension at http://www.temporal-analyst.com/

remediation strategies at contaminated industrial sites, or at regional scale, as in environmental management and resources supply studies.

From a wider perspective (Fig. 5), the prototype could provide storage and visualisation facilities for groundwater flow and transport models as well, addressing, among others, specific steady-state and transient calibration issues, as comparison of computed vs. observed data.



Arnold and Luinstra, 2009 Kresic, 2009

Fig. 5 – Database and GIS role in environmental data management and modelling

Most conceptual and numerical modelling environments, although providing advanced editing and visualisation features (Fig. 6), still lack of mature database support and, mainly for performance reasons, privilege the adoption of proprietary binary file formats. Examples include proprietary Hydro GeoBuilder[17], for model conceptualization, FEFLOW, a 3D finite element numerical code for flow[18], transport and heat simulation (Diersch, 2005; DHI-WASY, 2010), as well as OS finite difference numerical codes, as MODFLOW and SEAWAT[19]. All these systems provide their own pre- and post-processing environments (i.e. FEFLOW) or are embedded within third-party modelling environments (i.e. GMS, Visual Modflow[20]), but they would highly benefit from a better integration in the IT mainstream.

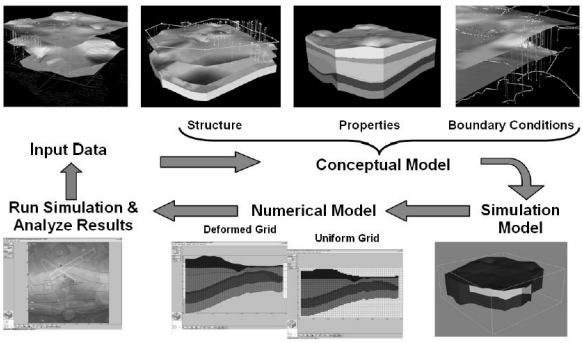
---

17 http://www.swstechnology.com/pdfs/technology_sheet/HydroGeoBuilder-Software-Spotlight.pdf

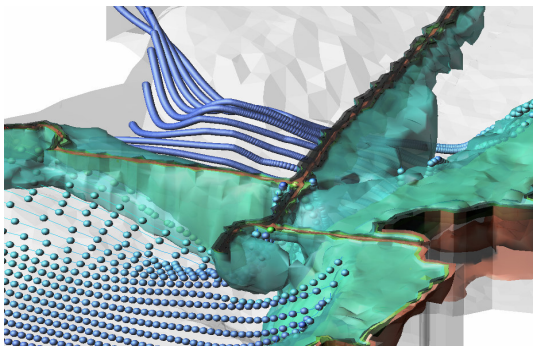
18 FEFLOW also supports flow simulation in density-dependent conditions, to simulate behaviour of only partially miscible liquids, as is the case for fresh and more dense salt water along coastal areas, leading to salinization phenomena

19 http://water.usgs.gov/ogw/seawat/

20 http://www.ems-i.com/GMS/GMS_Overview/gms_overview.html
http://www.visual-modflow.com/



Chmakov et al., 2009



Originally Tecplot (2011) designed after a FEFLOW numerical simulation

Fig. 6 - Information flow in building complex groundwater numerical models and advanced visualisation of 3D density-dependent flow near a coastline

Lessons-learned are expected to be relevant to any environmental domain, claiming for improvement in data quality management and overall effectiveness at both interpretation and modelling analysis stages, also thanks to effective visualisation. While most of these issues are already addressed by specialized geodatabase models and related geoprocessing tools, (Strassberg and Maidment, 2004) and partly addressed in my previous research and applications (Crestaz, 2003; Cascelli et al., 2005), the attempt is here to address the problem from a wider OS perspective, focusing on general purpose spatial database architecture and fully exploiting specific PostGIS inheritance features.

STATE-OF-THE-ART REVIEW: FREE AND OPEN SOURCE SOLUTIONS FOR SPATIO-TEMPORAL APPLICATIONS

Free and Open Source Software for Geomatics

Free and Open Source Software for Geomatics, hereafter referred to as FOSS4G, has been emerging in recent years as a rich valuable and reliable option against proprietary solutions for developing spatially-aware applications (Sanz-Salinas and Montesinos-Lajara, 2009; Steiniger and Bocher, 2009).

Although with a slightly different bias, free and open source software[21] has been around for long time, addressing non-spatial requirements in such different application domains as office automation, graphics, scientific analysis and advanced visualization. Examples[22] include Open Office, GIMP, R for statistical analysis, ESTAT (2010; Robinson, 2005[a,b]; Robinson et al., 2005[a,b]) for ESDA purposes, promoted by leading associations as Free Software Foundation (FSF, 2010) and Open Source Initiative (OSI, 2010). Opposite to proprietary software (not to commercial software, following a still common misconception), FOSS has been addressing the 4 fundamental freedoms, as stated by FSF (Steiniger and Bocher, 2009; p. 4):

- run the program for any purpose;
- study and adapt program at your own needs;
- redistribute the software;
- improve and release the new software for the benefits of the community.

[21] According to Steiniger and Bocher, 2009, OS would not necessarily cover the rights of modification and redistribution, as stated by FSF with reference to the concept of Free Software

[22] Programs information and source code can be accessed at:

http://www.gimp.org/
http://www.r-project.org/
http://www.geovista.psu.edu/ESTAT/

At least two of above rights imply that source code, rather than precompiled binaries, is accessible[23].

In order to guarantee above freedoms, different licensing policies have been proposed, including GPL, LGPL and BSD[24], just to mention few of them, most relevant differences concerning the right to keep modifications private, to release changes under different licensing conditions and mixing OS with proprietary software (Tab. 1).

License	Can be mixed with proprietary software.	Modifications can be taken private and not returned to the public.	Release changes under a different license
GPL	No	no	no
LGPL	Yes	no	no
BSD alike (BSD, Mozilla, MIT)	Yes	yes	yes (limited for Mozilla)
Public Domain (no copyright)	Yes	yes	yes
proprietary license	-	no	no

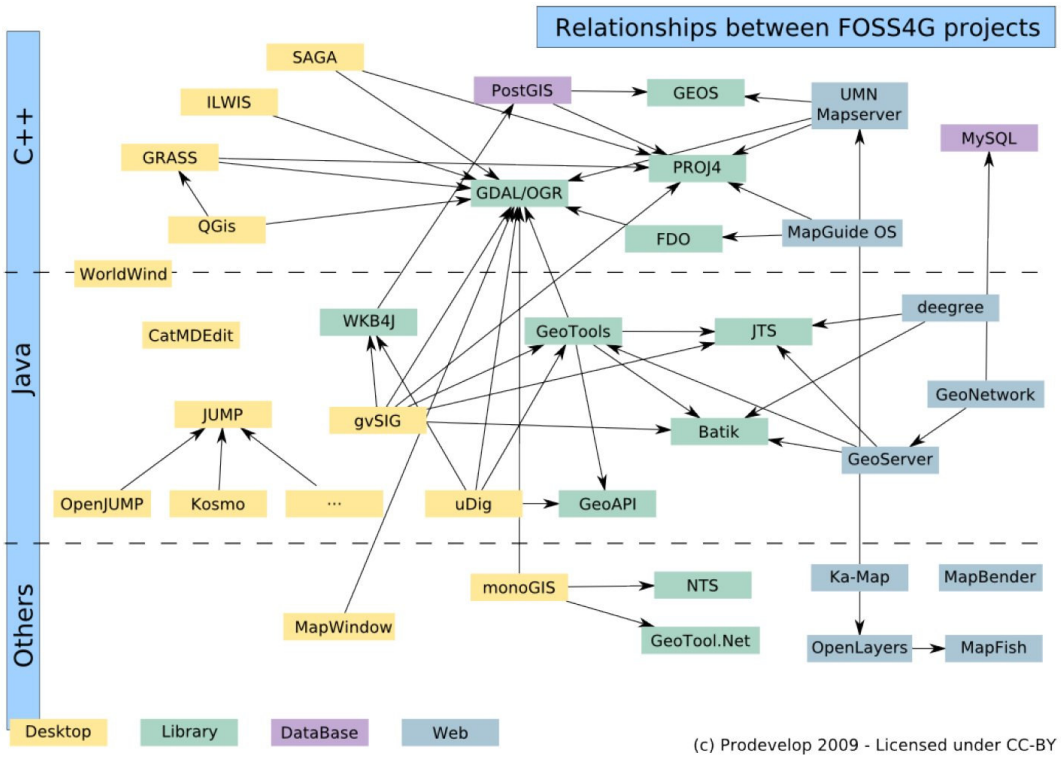
From Perens, 1999

Tab. 1 - FOSS and proprietary software: licensing terminology

Focusing back on FOSS4G, typical OS geospatial software stack include component libraries, which act as key building blocks, spatial databases, desktop GISs and GeoWeb Servers, resulting in complex relationships and dependencies, which are temptatively captured in Fig. 7, where programming language adds an additional classification axis.

[23] Shareware, although downloadable, must generally be registered and paid after a certain period of time. Still more important, source code can not be inspected, that's why it is generally classified as proprietary.

[24] For details on OS licenses, refer to OSI(2010) at http://www.opensource.org/licenses



Sanz-Salinas and Montesinos-Lajara, 2009

Fig. 7 - OS geospatial projects classification

Despite the framework complexity, a rough exam of figure already reveals, through the number of connections, how few projects are central to OS panorama, with a long history track, well documented and highly stable. Among them:

- Component libraries, addressing raster formats translation (GDAL/OGR, 2010), data projection (Proj.4, 2010), geospatial information manipulation (Geotools, 2010), 2D topological functions (JTS, 2010).

- Spatial databases[25], mainly dominated by state-of-the-art highly scalable PostGIS (2010), the geographic extension of ORDBMS PostgreSQL (2010), supporting OGC specifications as Simple Feature interface Standard (SFS). Integrated with both OS and

[25] MySql, despite its leading role in web applications and being recently added some spatial support, is not fully OGC standard nor complete and it is covered by a dual license. So it can not be regarded as completely OS and is not considered in current research.

proprietary desktop GISs (i.e. QGIS, ArcGIS through ArcSDE), PostGIS provides full support to geographic data and full featured geoprocessing functions. Few other projects aim at extending PostgreSQL/PostGIS functionalities, as: PL/R (2010; Conway, 2009, 2010; Obe and Hsu, 2010 ref. ch. 10), a stored procedural language integrating R statistical and graphing package within PostgreSQL, pgRouting (2010) addressing Location Based Services (LBS) issues as network analysis (i.e. shortest path computation), former WKT Raster, today known as PostGIS Raster, which will provide native raster support within next release of PostGIS 2.0 planned for April 2011 (Racine, 2010; PostGIS Raster, 2010).

- Desktop GISs, as QuantumGIS (2010), hereafter referred to as QGIS, for long time the only option to access PostGIS coverages and still today one of the most relevant ones[26], the Eclipse RCP (Rich Client Platform) based gvSIG (2010), originally funded by the Autoritat Valenciana in the framework of an internal migration project from proprietary ESRI platform, and uDig (2010), promoted by PostGIS developer Refractions Inc., and the more research focused OpenJump (2010) Java project. The latter offers few advanced features as the capability to issue geographic queries of any complexity against the database, visualize results and also save back data. These packages strongly differ each other, for installation easiness, features richness, documentation (users and developers manuals), geodata sources integration (i.e. with PostGIS, ESRI geodatabase) and support levels. GRASS[27] (2010), ILWIS (2010) and MapWindow GIS (2010) should be recalled as well, with their specific focus on environmental and hydrological data analysis. Analysis of OS vs. proprietary software and critical assessment of desktop GIS solutions advantages/disadvantages in Steiniger and Bocher (2009) and Steiniger and Hay (2009) address different perspective of users, developers and research (tab. 2). Chen et al. (2008) provide some guidance about

[26] However few annoying limitations of current QGIS stable version (1.5) exist, namely the impossibility to load data tables without geometry and the lack of multi-tables querying. Available plugins as PgQuery (http://spatialintel.blogspot.com/2009/03/leverage-power-of-postgis-in-qgis-with.html) unfortunately duplicate data, infringing elementary data management rules.

[27] By far the oldest and more powerful OS GIS platform, although only recently supported in native Windows mode. Its bad reputation to be particularly user un-friendly could be overcome as new QGIS steps in as its candidate GUI within the last version 1.5.0 (Thethys)

effectiveness of above platforms in poor hardware resources conditions[28], particularly referring to developing countries and their requirements in water management; quite interestingly, the analysis highlights how QGIS outperforms in some common tasks as satellite images[29] loading and panning, while other systems result in at least one order of magnitude higher reaction times (i.e. gvSIG) or even system crash (i.e. MapWindow GIS).

Task	GRASS	QGIS	ILWIS	uDig	SAGA	Open JUMP	Map Window	gvSIG	Arc View 9.3[e]
Viewing/exploration	●	●	●	●	●	●	●	●	●
Creation/ digitizing	●	●	●	●	●	●	●	●	●
Editing/updating	●	●	●	●	●	●	●	●	●
Conflation/integration	●			●		○			
Presentation									
Maps	●	●	●	●	●	●	●	●	●
Charts[b]	●	●	●		●	○			●
Plots[c]	Via R	●	●		●	○			●
Tables	●	●	●	●	●	●	●	●	●
Overlay analysis									
Raster	●	Via GRASS	●	Via JGrass	●	Via Sextante	●	Via Sextante	
Vector	●	●	●	●	●	●	○	●	Partly
(Spatial) statistics	Via R		●	Via JGrass	●	Pirol- JUMP	Raster only	Via Sextante	●
Customization (script or API[d])	API, Python, Perl	API, Python	ILWIS scripts	API, Groovy	API, Python	API, Jython	API (.Net)	Jython	Python and others
GPS data import	●	○	●	○	●	○	●	gvSIG Mobile Pilot	●

[a] ● Functionality provided, ○ functionality provided by software plugin (i.e. an extension). See also www.spatialserver.net/osgis/ for details.
[b] Charts: i.e. a thematic map that shows bar charts, pie charts, and graduated symbols.
[c] Plots: scatter plot, bar plot, histogram, etc.
[d] API: Application Programming Interface — a possibility for custom function development which enables tasks such as simulation and modelling.
[e] ESRI ArcGIS ArcView 9.3: we only assessed the standard functionality and not functionality that comes with extensions that require additional purchases.

Steiniger and Hay, 2009

Tab. 2 - OS Desktop GISs and ESRI ArcView 9.3 functionalities

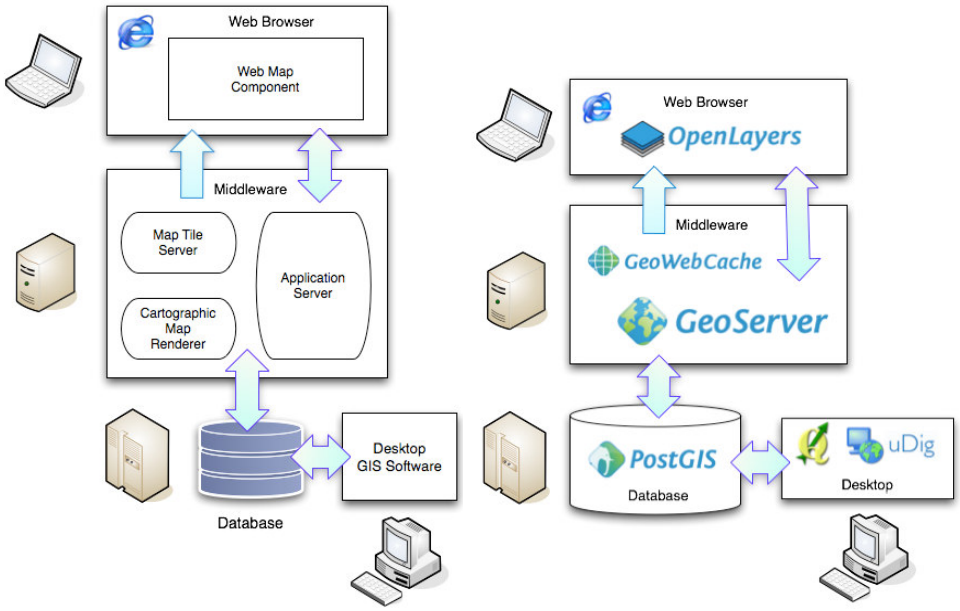
- GeoWeb servers, to serve geodata on the Internet. Long time established solutions include the NASA (2010), University of Minnesota (UMN) and Minnesota Department of Natural Resources (MNDNR) originated MapServer (2010) project, providing dynamically generated geographic web pages, and the JAVA2 EE Geoserver (2010), outstanding GeoTools project, optimal candidate, thanks to its support to WFS-T, for supporting enterprise requirements for true remote editing of geographic data.

Definitely an open source geospatial technological stack (Fig. 8) is a collection of layers of open source components or services used to provide a spatially-aware software solution or application. It tipically addresses complex requirements, as permanent spatial data storage,

[28] PCs 6÷10 years old, with RAM equal or less then 512 Mb, with MsWindows 2000 or XP Operative Systems.

[29] Landsat satellite images can now be download for free from http://glovis.usgs.gov/

desktop editing and querying, spatial analysis (Bailey and Gatrell, 1995; De Smith et al., 2009, Longley et al., 2005[b]), cartographic production, advanced visualization for ESDA and ViSC (see GeoDa example in **Fig. 9**), geospatial web applications development and also advanced spatial analysis applications, as is the case for the schematic geomorphological interpretation process (Fig. **10**) in SAGA (2011).



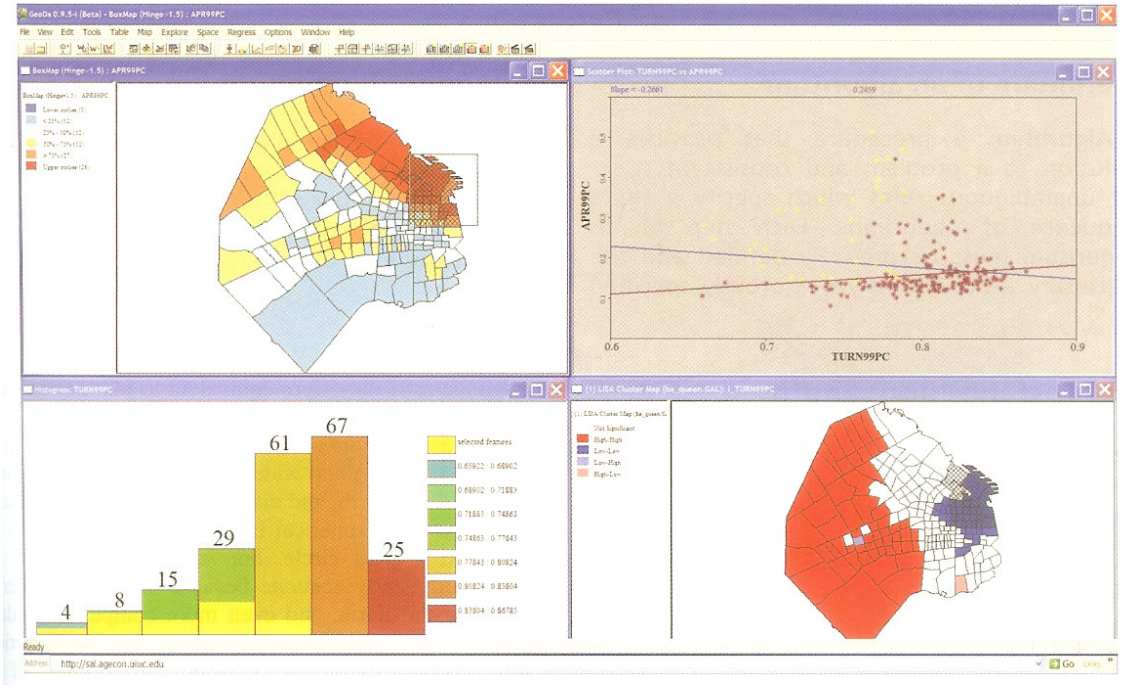
From OpenGeo, 2010

Fig. 8 - Geostack architecture: general framework and an OS option

The emergence of Google Maps API in 2005 (Williams, 2010) and its further OO developments (Google, 2010; Svennerberg, 2010), as well as other mapping APIs as Yahoo (2010), BING (2010), formerly Microsoft Virtual Earth, and the OS Open Layers (2010) opened and boosted entirely new fields of research and practice, as neogeography (Goodchild, 2007) and crowd sourcing by online communities (VGI), in the framework of new Web 2.0 revolution (O'Relly, 2005). Ushahidi (2010), originally conceived to support crowd violence reporting in 2008 Kenia post-electoral period, is a leading example of the latter, increasingly used worldwide in election processes, humanitarian crisis and environmental management monitoring.

Above APIs provide mapping components in the form of java script libraries, supporting client-side development of AJAX web applications, for integration – by means of open OGC compliant or proprietary web services - of background maps, remotely sensed images tiles or vector data, as well as user served data. Trends towards distributed geospatial processing is well highlighted by the new advanced services made available through the last release of Google Maps API:

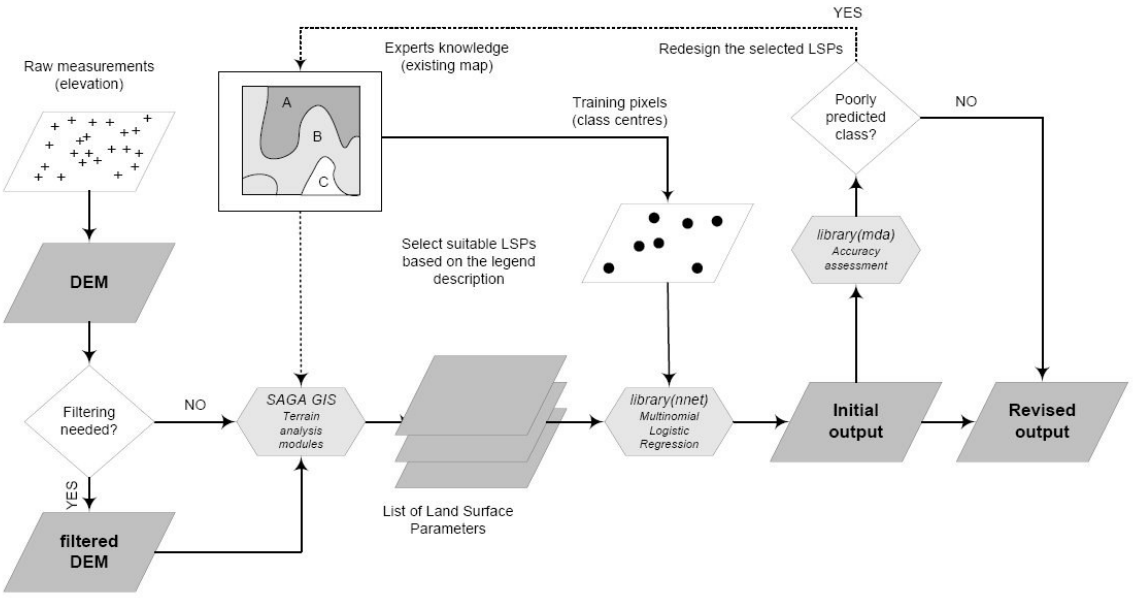
- Geocoding, returning point coordinates after textual address;
- Elevation, returning a DTM[30] value based on a global model after point coordinates;
- Direction, returning the optimal path connecting two input point locations, based on the specific transport mean.



From Longley et al., 2005[a]

Fig. 9 - An example of Exploratory Spatial Data Analysis in GeoDa environment

[30] DTM and DEM terms are generally considered as interchangeable (Burrough and McDonnel, 1986; Lillesand and Kiefer, 2000), although other authors emphasize differences between elevation in a built-in environment and the interpretation of the original surface. While generally not relevant in most Google Maps applications, this issue is central to processing of high resolution remotely-sensed images, as LiDAR.



Hengl, 2009 (ref. p. 215-218)

Fig. 10 - Hybrid expert/statistical approach to supervised geomorphological classification

Spatial databases

Object Relational DataBase Management Systems – Key Concepts

ORDBMS platforms extend traditional Relational Data Model (App. 1), fully exploiting foundation OO concepts (Connolly and Begg, 2010; pp. 921-970) of type (class equivalent), inheritance and associations, and providing a sound basis for the development of spatial databases.

Oracle provides a good example of extensive OR[31] support, addressing:

- Types (much like classes), used to model both state (data) and behaviour (methods[32]) of entities, leading to creation of tables of objects, instantiated after system or user-defined types;

31 http://infolab.stanford.edu/~ullman/fcdb/oracle/or-objects.html

32 Methods include default constructors, accessors and mutators (respectively to build objects and to access/set attribute values), as well as user-defined member, static and comparison methods. Method overriding should be supported as well, in order to address requirements for further specialization of inherited subtype (subclass) behaviour.

- Inheritance, supporting hierarchies through definition of both abstract (not instantiable) supertypes and specialized subtypes; both attributes and methods can be inherited by child classes from parent classes, while methods can be overridden to address requirements for further specialization of inherited subtype (subclass) behaviour;
- Associations, addressing relationships modelling, as 1-M or M-M, through nested tables[33], which mask underlying relational joins by introducing a navigational approach logic through REF and DEREF operators[34];

From the standpoint of software engineering, UML provide different approaches and tools (i.e. class and interaction diagrams; Ambler, N.A.) to investigate user-defined types, their associations/compositions, relationships (and multiplicity), contributing to make clear conceptual architecture, semantics and functional interactions.

Specific Oracle implementation issues, worth remembering, include:

- object tables based on user defined types[35]. As each object is assigned an implicit unique identifier (object ID, which, by the way, can be defined as coincident with a primary key or system generated), REF and DEREF SQL statements can be used respectively to define a pointer to a record in another table and to access record by deferentiation.
- different types of collections[36], i.e. a VARRAY for a phone numbers list, or a nested table, supporting storage of data related by a 1-M relationship within the table itself.

---

[33] General issues: http://www.orafaq.com/wiki/NESTED_TABLE
http://www.dba-oracle.com/tips_oracle_varray.htm
Performance issues in: http://www.dba-oracle.com/art_9i_data_model.htm

[34] Much like pointers and their memory deferentiation in C programming language, REF and DEREF Oracle operators support storage and direct navigation to referenced records in related tables, generally resulting in more expressive and human-readable SQL statements than their equivalent join expressions.

[35] SQL syntax being "*CREATE TABLE usertableName OF userType*"

[36] VARRAY is variable in size up to a stated maximum, each element being referred to by indexing (as in most programming languages); nested tables provide more flexibility to accommodate variable amounts of data, while satisfying querying capabilities requirements. See Connolly and Begg for a brief review of major concepts and syntax behind the definition of nested tables, including the hidden complexities to keep trace of 1-M relationships between objects in main and related nested table.

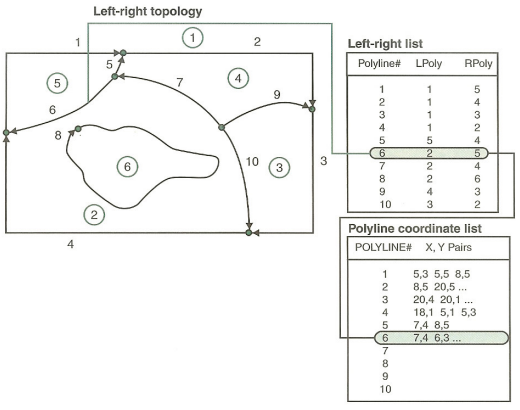
- REFs (conceptually similar to pointers in C and C++) providing support for navigation paths, against traditional relational approach based on joins after PKs and FKs.
- MAP FUNCTIONs, which, further to supporting specific user requests, also acts "*under the wood*" as a sorting function, addressing the specific requirements of clauses as ORDER BY in SELECT statements.
- the use of implicitly built-in constructors.

Despite its much less rich and powerful OR support, PostgreSQL provides an interesting feature known as table inheritance. Complex hierarchies can be seamlessly accessed through the top parent table, enabling development of complex table partitioning strategies. This approach has also some major caveats, as lack of constraints inheritance leading to extra coding efforts. Refer to Obe and Hsu (2010; ch. 3) for an in-depth review of table partitioning benefits vs. shortcomings. The issue is further discussed at spatial database prototype design and development stages.

For completeness, criticism about the OR model should be recalled as well. It would add complexity and overhead to the traditional pure relational model, further to lack of control over internal structures, while definitely still supporting storage to underlying relational tables[37]. Nested table solution to model M-M relationships highlights this additional (generally undesired) complexity, also if expressiveness of OO model in UML diagrams and the REFs navigational logic against traditional joins approach effectively maintain a great attractiveness.

Spatial databases: Key Concepts

Spatial databases are databases providing specialized support for storage, querying, retrieval and analysis of spatial data, that's data referring to locations on earth/planets surfaces, projected on planes or on arbitrary 2D space.

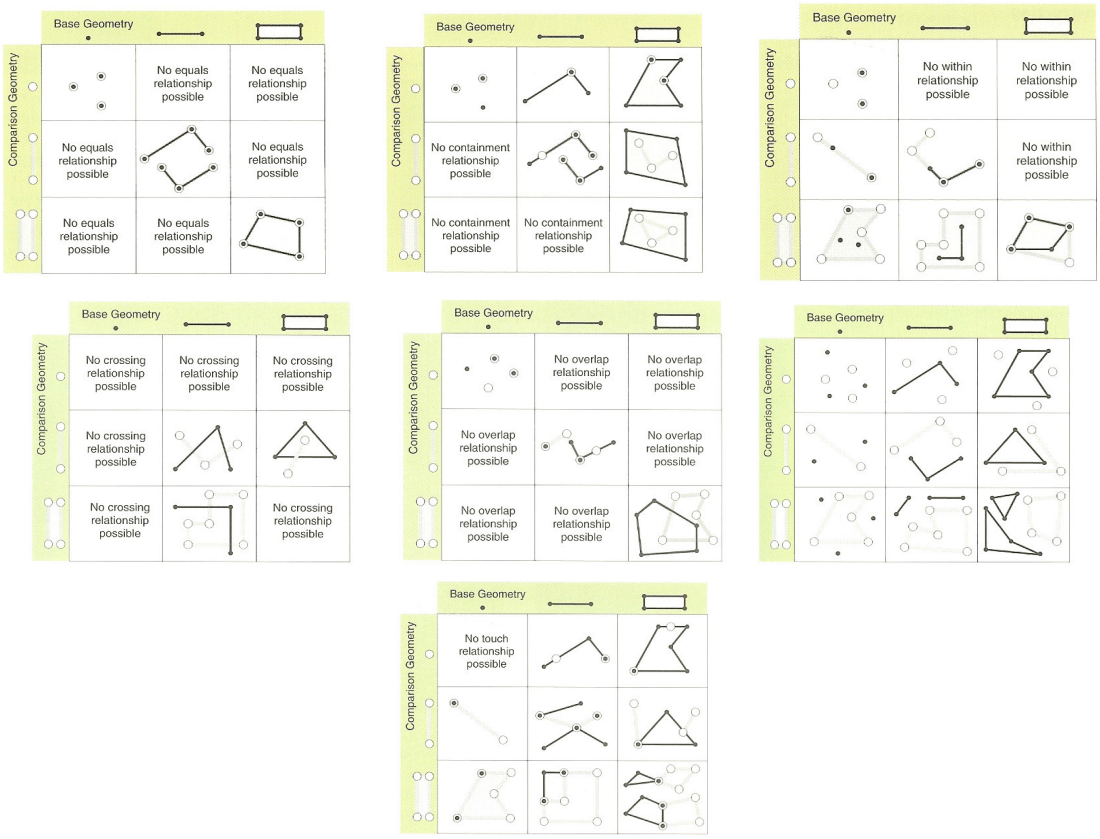


ESRI (1997)

Fig. 11 - ArcInfo topological vector geometry

[37] See http://asktom.oracle.com/pls/asktom/f?p=100:11:0::::P11_QUESTION_ID:2318607631616

Spatial data are intrinsically more complex than most native data types supported by traditional RDBMSs, as string of characters, numbers or dates; further to coordinates of points, polylines and polygons, projection systems, correctness (i.e. polygon boundaries cannot cross themselves) and topological[38] consistency (Fig. 11) must be supported as well, further to composite multi-geometries (i.e. multi-polygons) and specialized geographical operators (as operators addressing characteristic topological relationships; Fig. 12).



Zeiler, 1999

Fig. 12 - Topological operators

Dimensionality higher than 2D, as 3D and time-dependent issues, and geographic coordinates on earth spheroidal surface, further to projected coordinates, must be supported as well.

[38] Topology is the mathematics discipline concerned with spatial properties preserved under objects deformation (http://en.wikipedia.org/wiki/Topology). Early examples include ESRI ArcInfo and more recent developments in geodatabase technology (ESRI, 2003).

Spatial data are generally huge, for long time imposing specialized processing within binary files kept distinct from attribute data stored in tabular format, following the so-called dual-architectures pattern (i.e. shape files; ESRI, 1998). As spatial data querying and processing can rapidly turn to a nightmare as number of geographic features increase, traditional binary search algorithms have been complemented by the idea of space partitioning, at increasing levels of complexity and effectiveness, including (Fig. 13):

- regular grids, unlikely resulting in storage overloading with strongly spatially clustered data and inherent difficulties to choose optimal grid size;
- quadtree, following a tree-like space partitioning, differently refined depending upon data granularity and spatial distribution;
- RTree, addressing space partitioning through irregural rectangles, algorithms nowadays supported by leading proprietary and OS spatial DBMSs, as Oracle and PostGIS,

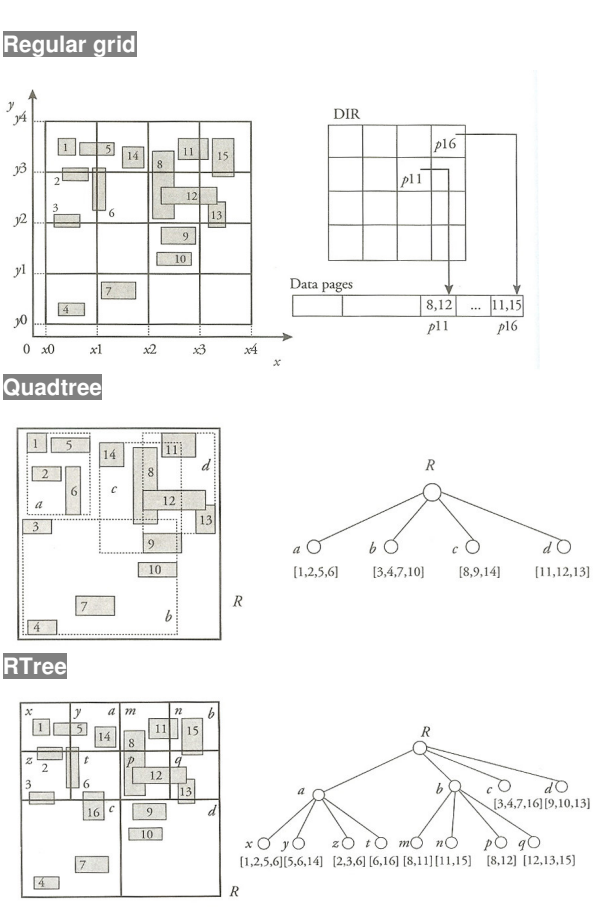


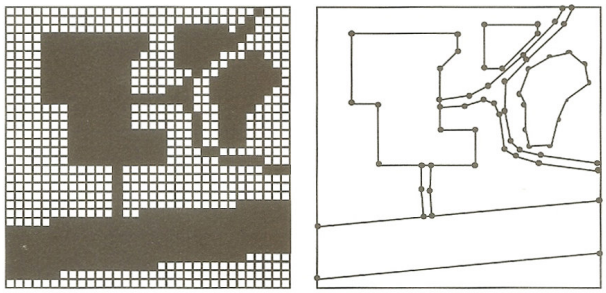
Fig. 13 - Different spatial partitioning schemes for indexing geographic information

PostgreSQL and PostGIS

PostgreSQL has been developed as OS in last 15 years, really dating back to Stonebraker's research at Berkeley Un. and promoting state-of-the-art OO concepts. It supports standard database types (i.e. integer, text, date), is fully ANSI:2008 compliant, consistent with OGC specifications and highly scalable. It is packaged with a user-friendly management and querying environment pgAdmin III (conceptually similar to systems provided for MySql and Oracle).

Relevant to current research, the geographical PostgreSQL extension PostGIS is today provided as an option in standard database installer, addressing both basic requirements for effective geodata storage, although with a major bias towards vector model (Fig. 14), as well as simple and advanced geoprocessing requirements. Specialized geographic data types (geometry and geography, the latter being relevant to modelling locations over the spheroidal earth surface) are integrated by specialized RTree indexes, for effective querying.

Much like Spatial Oracle in the proprietary software arena, PostGIS has taken for free hundreds of OGC compliant spatial functions[39] to the wide public of the OS community, while being recently (since ArcGIS 9.3) integrated also in the commercial ESRI stack. Functions range from 'simple' geometric computation, as length, distance and area, to more complex topological tasks involving containment, intersection, union and difference, just to recall few of them.



Worboys, 1995

Fig. 14 – Raster vs. vector spatial data model

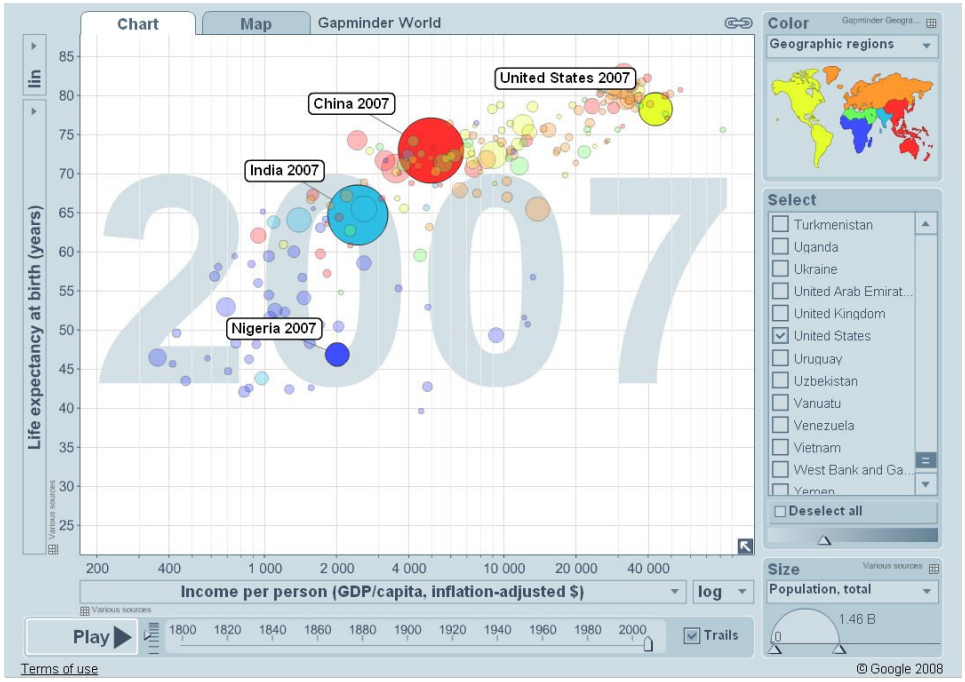
PostGIS features are fully accessible through SQL/MM, supporting the migration of most programming and analysis logic and tasks from desktop GIS to the database, with increased concerns for enterprise server side control, efficiency and security.

For most people, geographic data make sense only on a map, provided that also the research community has widely recognized the value of maps for ESDA scopes (Kraak, 2005) and effective communication of quantitative data (Tufte, 1997; 2001; example in **Fig. 15**). That's why

[39] Actually part of the functions have been duplicated, in order to adhere to standards without loosing backward compatibility. Interestingly enough, ST prefix for new functions (i.e. ST_Length, ST_Intersection) stays for Spatio-Temporal, pointing at the perceived relevance of coupling the two dimensions. Note also that Spatial Oracle is sold as an extension of top-level Oracle Enterprise, while very few geoprocessing functions are made available in free entry-point Oracle XE or other priced licenses, under the common brand of Locator (Ref Jeff Hobbs blog at http://jeffhobbs.net/2007/09/24/oracle-locator-vs-oracle-spatialdo-you-know-the-differences/). Geocoding and network analysis are not supported as well.

the integration of PostGIS within most of the OS desktop GISs, as well as the proprietary ESRI platform, is particularly relevant to its wide adoption and diffusion.

Definitely compliance to SQL standards has much to do with interoperability, that's the ability of "*two or more information systems to share information or processing capabilities*" (Worboys and Duckman, 2004; p. 259) and to advancements in SDI design and development.



Gapminder, 2011

Fig. 15 - Socio-economic data analysis using Google Motion Chart

SPATIO-TEMPORAL APPLICATION: GENERAL ISSUES

General framework

Current chapter presents general development strategies and requirements analysis for the environmental spatio-temporal database application design.

Based on open source database technology, the application addresses requirements for effective concurrent centralized data management. The prototype has been developed in PostgreSQL/PostGIS, addressing the requirements for a generic architecture coping with both spatial and temporal dimensions and investigating its object oriented nature, namely its table inheritance features.

At prototype core, the spatial database is intended to minimize risks of data duplication and integrity failure, guaranteeing effective storage and querying of monitoring geometries, whether simple (i.e. point locations) or complex ones (polylines and polygons), measurement types and related time series.

Users are expected to interact with the spatial database through different tools, depending upon their own background and specific needs, by direct database connection or by consuming web services, in the framework of desktop GISs, advanced visualization tools or specialized web mapping or web GIS applications. Different options are briefly summarized .

As a final proof of concept, a web mapping application has been designed and developed, to provide evidence of effectiveness of server-side access to spatio-temporal data as well as user interaction on browser client side. At an early stage of analysis, potentials of the OS java script framework GeoExt (2010), which includes mapping OpenLayers and rich content ExtJs (Sencha, 2010) libraries, have been investigated. However, Google Maps API v 3 has been preferred to develop the final prototype, basically due to its Object Oriented architecture, higher programming flexibily, better documentation and, last but not least, the availability of various Google time visualisation flash widgets particularly useful at integrating the spatial and temporal dimensions. Same programming concepts would equally apply to Open Layers application development.

Google Maps API is not open source but it is free, given few restrictions, as the application must be freely accessible on the internet, 'Google' trademark logo not removed and web traffic not 'massive'. With the new API release, a user key is not requested any more.

User group(s)

Different stakeholders interested in environmental issues, ranging from advanced users focused on a consistent and effective approach to spatial data management and modelling, as database experts, spatial analysts and modellers, to decision makers looking for integrated approaches and tools to support policy making process, to large public, i.e. in the framework of PPGIS (Public Participation GIS) applications promoting democratisation efforts.

Experts are expected to come from different domains, as international organizations, NGOs, governments, private companies, addressing different issues as climate change, environmental contamination and flooding, just to recall few of them. Although at totally different level of complexity, top executives, technical managers and large public are all expected to benefit from the proposed approach, accessing information through database managers, specialized GIS or dedicated web mapping applications.

Expected benefits and suitability assessment

Proposed approach brings all benefits of any centralized spatial data management architecture, including improved quality, documentation (metadata; FGDC, 2010) and security, while guaranteeing efficient concurrent access to spatial data and GI (Geographic Information).

Although few proprietary architectures, as state-of-the-art ESRI ArcGIS Server coupled with enterprise Oracle RDBMS, are currently available in the market, still major financial and technical constraints strongly limit their wide adoption and acceptance. Except for large companies focused on cartographic production and, at a much less extent, spatial analysis, the prevalence of poorly structured, mostly file-oriented (i.e. shape files) and desktop GIS approaches to spatial data management and analysis generally leads to inconsistent practices, data integrity risks, knowledge and economic loss.

OS technology is nowadays a good candidate to a mature, open-standard and cost-effective approach to spatio-temporal data management, particularly the postgreSQL/PostGIS platform

addressed by current research. Spatial database can be easily accessed from OS desktop GISs, spatial data delivered through spatial web servers and integrated in web mapping mashup applications (Batty et al., 2010).

Complete assessment of proposed OS technology should include other financial and management constraints, as the additional costs for hiring more-trained IT personnel, the requirement for long-term support, and the possible bias towards adoption of industry-standard software in critical mission projects (Igualada, 2009; Stewart et al., 2009). The limited percentage incidence of GIS software license costs vs. overall budget in typical mean and large scale projects (generally below 15% according to Maguire, 2005) should be taken into account as well.

Key application requirements

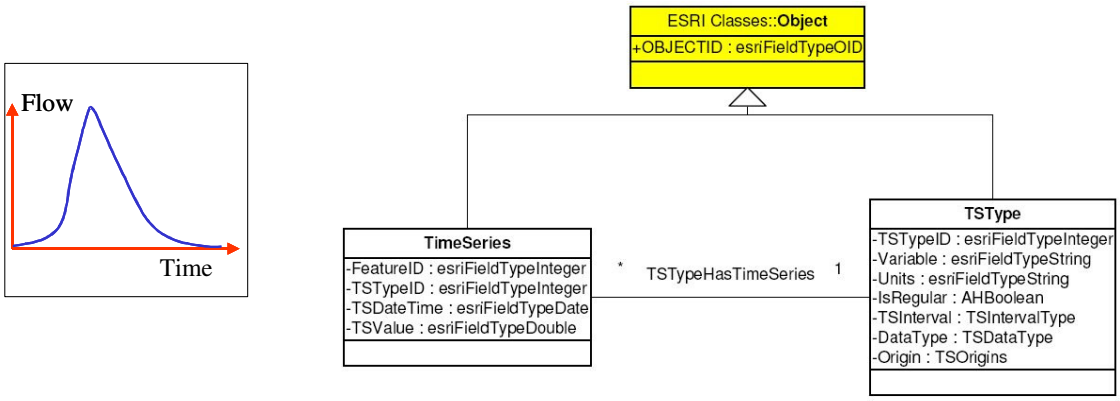
Proposed OS spatial database architecture is designed after guidelines provided by ESRI Hydro data model (Maidment, 2002; CRWR, 2010). This model, despite being focused on surface waters, reveals a high degree of flexibility in accommodating spatial data and time-dependent monitoring data of arbitrary type[40]. A screenshot of relational database architecture (Fig. 16) shows the three key tables, fields and underlying relationships, namely:

- monitoring point table, storing unique ID, point code and location[41];
- measurement table, storing unique ID, date and value, further to two foreign keys maintaining references to location and measurement type tables;
- measurement type table, storing unique ID, variable name and measurement units, further to other fields supporting finer type description.

Unique IDs in each table act as primary keys.

[40] The model can flexibly accommodate both recorded and generated data; so modelling output could be temporary stored to database, i.e. for calibration purposes by comparison with monitored data.

[41] ESRI shape field can include both 2D coordinates and elevation. The latter has not been explicitly addressed in proposed application, while in specific applications, as in subsidence studies focused on variation of elevation vs. time, elevation should even migrate to measurement table.



CRWR, 2010
Fig. 16 - Hydro data model: key tables schematic architecture

Database is expected to be accessed through a geoweb application, providing cartographic background, standard navigation tools (i.e. zoom, pan) and time series visualization features, based on user selection of variable type and monitoring points of interest.

Major non functional requirements include:

- a minimalistic cartographic full screen layout design, populated with background and user defined geodata;
- specialized tools to load and visualize time series.

User Interactions

User interactions and main (human and not human) actors are captured in the system use case diagram here below (Fig. 17), which include:

- generic application users (domain experts, executives, wider public);
- administrator, a specialization of the generic application user in charge of data loading to spatio-temporal database;
- geoWS providing background maps and support to monitoring points visualisation and users interaction;
- clustering Web Service, aimed at effective visualisation of large data sets;
- exploratory spatial data analysis, focused on time series analysis in the specific case.

An expanded description of '*Browse geodata*' use case captures details of typical user interaction with web mapping application.

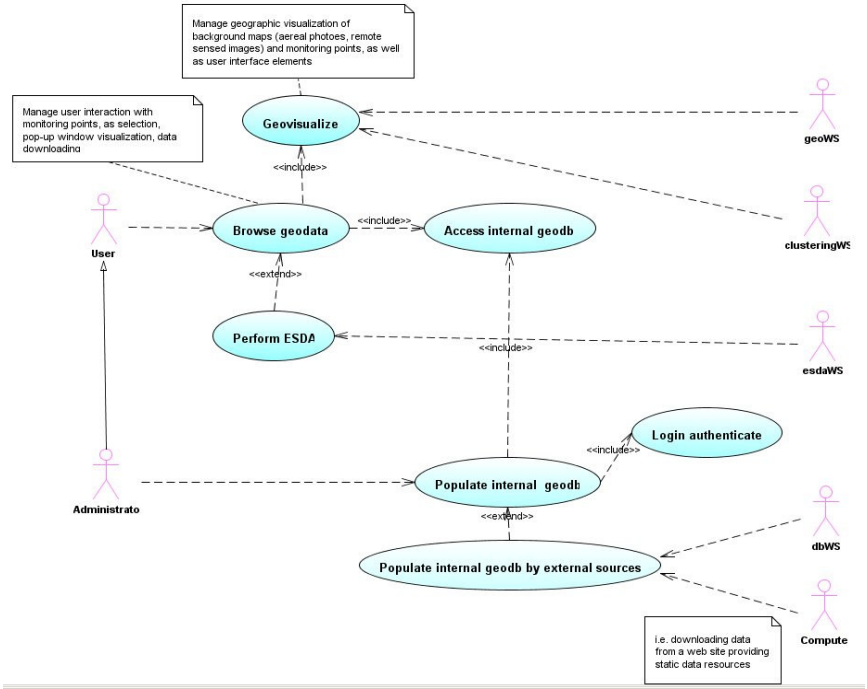


Fig. 17 – Use case diagram for the web mapping application

Use case: Browse geodata
Actor(s): User
Goal: Surf through the cartographic layout of the web application, query geodata by location and/or perform exploratory spatio-temporal data analysis on one or more time series related to monitoring points, to spot underlying statistical relationships and/or patterns by visual inspection.

Overview:
User(s) navigates cartography by zooming in/out and panning, based on geoWS functionalities, selects one or more monitoring point(s) by location to access related time series, stored to application spatio-temporal database, and loads time series to a specialized widget providing basic exploratory tools.

Cross reference:
-

Typical course of events:

	Actor action		System response
1	Zoom in and/or pan to locate area of interest	2	Update cartographic view
3	Select monitoring point(s) to access basic information	4	Update cartographic view highlighting selected point(s), each time updating a pop-up window with relevant information
5	Press "*Time series*" button to access time series for currently selected variable and monitoring point(s)	6	Query spatio-temporal database and load time series to Google time annotation widget
7	User conducts time series analysis (i.e. zoom in, pan, etc…)	8	Google time widget provides all functions already built-in

Alternative courses:
-

Fig. 18 - Use case "Browse cartography": expanded description

SPATIO-TEMPORAL APPLICATION: DESIGN AND PROTOTYPE DEVELOPMENT

Database design guidelines

Based on state-of-the-art geodata models in environmental sciences (ESRI, 2010), traditional RDBMS applications in water data management (Karanjac, 1993; Gocu et al., 2001), and particularly the ESRI hydro and groundwater data models (Maidment, 2002; Strassberg and Maidment, 2004), a general-purpose spatio-temporal database schema focused on environmental monitoring data management has been designed.

Investigation has been focused on an effective strategy towards storage, querying and retrieval of environmental geographic information (GI) and related temporal series, including both physical and virtual data (i.e. collected at monitoring stations, resulting from numerical simulations), specifically addressing multi-temporal system requirements (Langran, 1992; Frank, 1998; Ott and Swiaczny, 2001; Voigt, 1998 for time-management in groundwater modelling).

Particularly, the ESRI-CRWR ArcHydro data model provides a very effective and elegant object oriented approach to the problem. A generic geographic table (Fig. 19) acts as a parent table in the hierarchy for specific geometric tables, which can be further specialized to capture a specific application domain ontology. Wells, rivers and lakes provide relevant examples in the framework of an hydrological application.

A time series type table (tsType) provides storage for each variable type, including, among others, measurement units, reference symbol and origin, namely measured (recorded) or computed (simulated). An example would be “Piezometric head”, “m a.s.l.”, “h”, eventually set as recorded (a coded value or a boolean). This approach leads to some redundancy, as two distinct records must be added to point at the same measurement type addressing observed and computed data; such a redundancy is perfectly acceptable, as a result of a denormalization choice.

Actually an intermediate table (ts) provides storage for time series data, namely date/time and value, plus the two foreign keys which point at the monitoring location and the measurement type series.

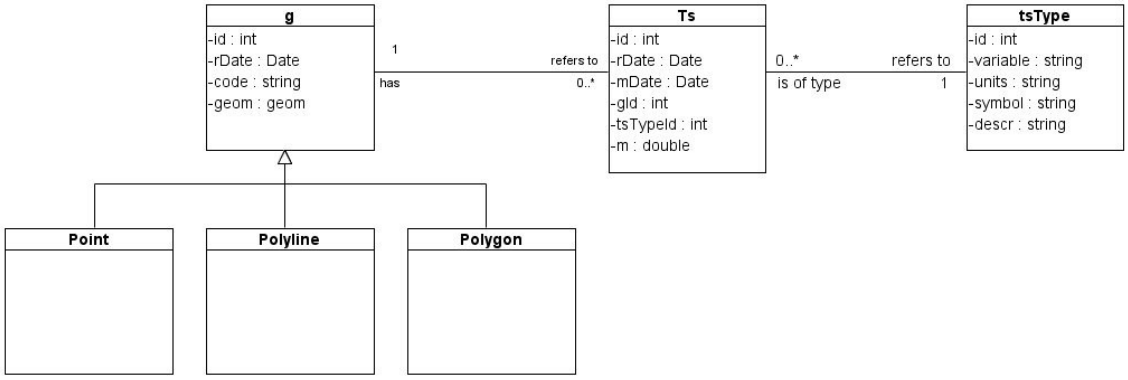


Fig. 19 - UML data model for spatio-temporal data management: generic conceptual scheme

PostGIS database design

Based on above guidelines, the spatio-temporal database design has been refined with specific focus on PostGIS and its table inheritance implementation (Obe and Hsu, 2010; par. 3.2.3).

Table inheritance has been used to partition geographic features by geometry type. Provided that the geometry parent table acts as a unique entry point for insertion and querying operations, spatial data are actually redirected to child tables through triggers based on geometry type criteria.
Each geographic feature is uniquely identified within the spatial database by an id, conceptually equivalent to the Hydro Data Model HydroID (ESRI, 2005).

User is expected to access data indifferently through the top parent table or child tables, depending upon specific requirements and/or concerns (i.e. faster querying). Advantages of the later approach include:

- limiting the overhead of cross joins, which would result from accessing different geometries from the same spatial table. A simple example, involving spatial intersection, would point at selection of monitoring points within specific sub-basins;

- easier data migration to/from formats/systems which do not support multiple geometries within a spatial table (i.e. ESRI shape file).

PostGIS table inheritance is very powerful, yet suffering from some relevant drawbacks compared to true OO Oracle implementation, including limited expressiveness (i.e. no Types, VARRAY or REFs) and lack of constraints inheritance. At least this last issue can be addressed through SQL triggers, enabling complex relationships maintenance behind the scenes.

So PostGIS data model design (Fig. **20**) somehow diverges from the ideal case study, imposing relationships duplication at lower levels of the inheritance hierarchy, leading to a rather inelegant yet pragmatic and effective solution. Most important, user is potentially unaware of what is going on behind the scenes, while benefiting from referential integrity constraints and the unique identifier shared by all geometries.

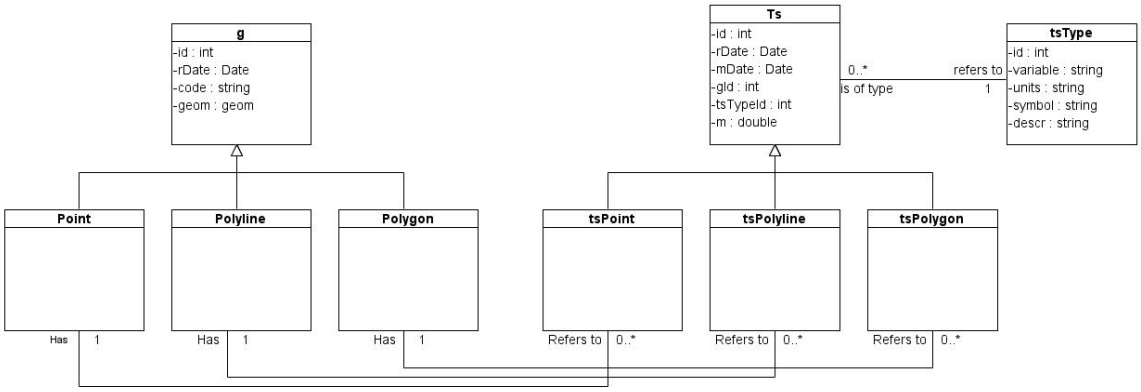


Fig. 20 - UML data model for spatio-temporal data management: PostGIS design based on table inheritance

PostGIS database implementation

The PostGIS spatial database component of the Environmental Management System (hereafter referred to as EMS) has been developed and tested in SQL, using PostgreSQL 8.4.4 and client pgAdmin III, v. 1.10.2

EMS can be accessed through any GIS client supporting PostgreSQL connection, based on parameters provided in Fig. 21 for Quantum GIS. People could equally recreate the spatial

database on their own server or on local computer (local host), provided that parameters are modified consistently.

SQL source code is available in app. 2 and fully commented. Most relevant points are commented here below, particularly addressing non-trivial issues:

- Check of PostgreSQL and PostGIS installed versions and creation of the ems schema, where all other database objects will be created.

- Creation of the geographic parent table g. The basic idea is that table g will act as a virtual table, providing the entry point to insert and query spatial data, which will be migrated to child tables based on geometry data type. Attributes include:

  1. the automatically generated (serial) id, acting as a PK, inherited by features in child tables, acting as a unique reference at database level to link time series. The id acts much like the HydroID in Hydro Data Model.
  2. the recording date, rDate, enables users to keep trace of when information has been made available, This has potential drawbacks particularly for environmental legal actions; typical questions could address if you knew about contamination at a given time or how much time has passed in between sampling and making data available. Same concept applies to all other tables.

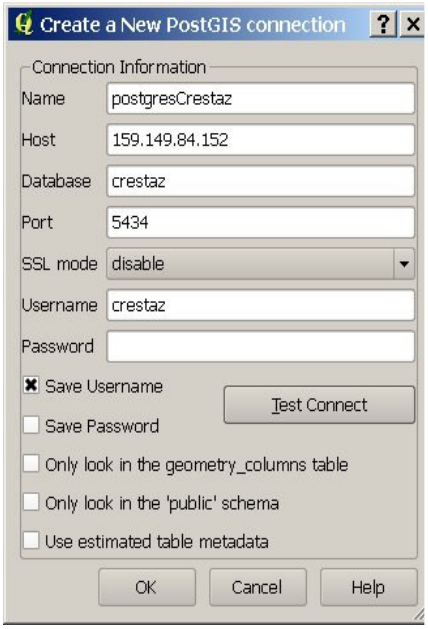


Fig. 21 – PostGIS connection window

  3. the geometry identification code. While it should be unique in the framework of a small project, however the same code is generally repeated over and over by different actors, in legacy databases and bibliography. In these cases we can not rely on a simple unicity code constraint and more reasonable geographical constraints, imposing that different geometries do not share same code and space, give rise to other issues in spatial reasoning, involving intersection and spatial

resolution. Tipically assessment of different point geometries coincidence is not a trivial task, involving spatial quality, scale and resolution issues in order to estimate what is the minimum distance which makes sense. Another problem is related to the bad, yet common practice, to re-drill abandoned or damaged boreholes inheriting original code; this often occurs in the framework of local scale environmental characterization or remediation projects, where monitoring point code is often so strictly tied to a location and to related contamination issues that also environmental authorities privilege such an approach.

A generic geometry attribute is added only at a later stage, once child tables have been created. An attempt to define it at this stage would conflict with geometry data type constraints in child tables.

- Creation of the specialized geographic child tables, point, polyline and polygon, addressing storage of features based on geometric type criteria. Provided that multi-points or multi-lines features can be easily splitted to their simple counterparts, while complex polygons, as a polygon with a hole or a multipolygon resulting from a spatial operation, can not be further simplified, only multi-polygons are explicitly supported. Tables creation is straightforward, inheriting from parent table g. Unfortunately PK constraint must be redefined on inherited attribute id, as PostGIS implementation does not support constraints inheritance. Geometry attribute '*geom*' is added through the AddGeometryColumn function, which is a good practice and key to access from most desktop GIS tools[42], as Quantum GIS. All geometries are assumed to be in WGS84 geographic reference system, as stated by the EPSG (2010) SRID 4326. If projected data should be migrated to the database, PostGIS functions as ST_Transform can be used to convert spatial data to WGS84, as later demonstrated in the case study. Spatial data type and 2D constraints are equally added through the AddGeometryColumn function.

[42] PostGIS provides a system table named geometry_columns, stored to public schema; this table keeps trace of any metadata about spatial tables in the database and it is generally accessed by most GIS tools to retrieve information about spatial reference system (SRID), dimension and geometry type. Specific PostGIS functions take the responsibility to keep updated the spatial metadata table, as: AddGeometryColumn, DropGeometryTable, UpdateGeometrySRID and Populate_Geometry_Columns

- Creation of spatial GiST indexes, on geometry fields, and binary tree indexes, on code attribute, in order to guarantee fast access to records, including the addition of the generic geometry attribute to the geometry parent table g.

- Creation of a table, gRejects, sharing the same structure of the 'g' geographic table (although staying apart from the inheritance tree) and acting as a bin to store features with a currently unsupported geometry. Applications could access bin content, for example after an INSERT operation, to show users what records failed to load and possibly why. At the moment, database prototype does not further integrate/specialize this table, but at least a further attribute could keep a description of problem nature, as unsupported type or duplicate record (see previous analysis on geometry duplication).

- Creation of a rule chkins_g, which intercepts spatial data insertion attempt to parent table g and redirect to specialized child tables, based on geometry type, or to gRejects table, at the moment for unsupported geometry types. Data insertion testing of both supported and unsupported geometry features reveal, among others, dispatching strategies and unique ids generation at database level.

- Creation – analogously to previous sequence - of a time series parent table storing following attributes:
  1. an automatically generated (serial) id, acting as a PK;
  2. a FK, gid, pointing at reference geometry (i.e. a measure taken at an air monitoring station);
  3. a FK, tsTypeId, pointing at reference measurement type (yet to be defined at this stage);
  4. a recording date, rDate, to be automatically populated at insert time;
  5. a measurement date, mDate;
  6. a measurement, m.

- Creation of child tables tsPoint, tsPolyline and tsPolygon, to store time series related to specific geometry type features, PK constraints and binary tree indexes on time. Creation of the tsRejects table, which, aimed at storing records violating unicity constraints, stays apart from the inheritance tree. This is a quite inelegant solution, but unfortunately PostGIS table inheritance implementation does not permit to refer back to

geometry PK through the parent table g. As spatial data are migrated to child tables, time series data must be migrated to their own child tables as well.

- Creation of a rule, chkins_ts, to redirect records to specific child tables or to tsRejects table. Although similar to previous rule chkins_g, this rule is just a bit more complex. It traces back to referenced geometry through the FK gid, in order to assess its geometry type and take the appropriate path through the inheritance tree. FKs and cascading behaviour settings are defined at this stage. A series of insertion tests demonstrate the effectiveness of proposed solution, including errors due to reference to non existing geometries.

- Creation of general purpouse domains as dmDataType and dmOrigin. The former points at the measurement general meaning, namely if it refers to instantaneous, cumulative, incremental, average, maximum or minimum data, particularly relevant in case of automatic data recording instruments providing some sort of aggregation. The latter simply points at the nature of the measurement, namely if it is observed or computed, as it would be the case for a numerical simulation scenario.

- Creation of time series type table, tsType, including following attributes

  1. an automatically generated (serial) id, acting as a PK;
  2. measurement type code, i.e. ‘T’;
  3. measurement variable long description, i.e. ‘Temperature’;
  4. measurement units, i.e. ℃;
  5. data type, based on dmDataType domain;
  6. data origin, based on dmOrigin domain.

  Unicity constraints are added at this stage. The table is somehow unnormalized, as at least two distinct records would be required to refer to the same measurement for observed and computed data. Apart from the PK, the two records would be basically distinct only for the different codes. Anyway this is a compromise which, consistently with the Hydro data model, looks perfectly acceptable in this framework.

Once again, FK constraints are defined on tsPoint, tsPolyline and tsPolygon tables, referring to tsType table PK, including specific cascade rules. Insertion examples on both tsType and ts tables document the effectiveness of proposed approach.

- Creation of three views, pointView, polylineView and PolygonView, to enable fast and efficient access and query through desktop GISs. Particularly QuantumGIS requires that a 4 bytes (int4) PK in an underlying table is included in the view, to be used as a unique record reference. The id in ts table is included for this reason. Views are also explicitly registered in the spatial metadata table geometry_columns, due to the additional QuantumGIS requirement already commented above.

Above architecture leaves open the door to any further specialization of the inheritance tree, following a specific application domain ontology.
For example in the context of an hydrological application, real world objects as wells, river gauges, channels and basins could be refined and possibly enriched with modelling-oriented features for later coupling with numerical modelling codes.

Spatio-temporal web mapping application

The web mapping application has been developed combining basic HTML, and CSS for basic web page design and styling, client-side java scripting (app. 3), accessing last release of Google Maps API v. 3 for mapping components, google flash widget AnnotationTimeLine[43] for spatio-temporal dynamic data charting and PHP server-side scripting (app. 4) to access spatio-temporal data.

Key programming techniques and patterns have been grasped from Svennerberg (2010), taking into account the OO asynchronous architecture of Google Maps API.
Various java script libraries are loaded in HTML, as the Google Maps API itself and the fluster2 library, supporting run time grid clustering of monitoring point locations.
Type series list and buttons, on top left browser window immediately above the map and chart divs, provide access to application specific functions,.

[43] See eVis (2010) for temporal series management in desktop Quantum GIS

Key source code, in map.js library, is fully encapsulated in an outer anonymous function, assigned to the window onload event and providing following key features:

- Integration of Array class with a contains method, later used for management of markers selection;
- Definition of markers images (and shadows), based on “monitoring point” standard symbol in UNESCO (1970) hydrological legend;
- Map creation based on HTML div, accessed through native DOM getElementById function, and a map options object literal providing standard configuration, namely mandatory variables map center, zoom level and type, further to other optional variables as scale control or dragging cursor icon;
- XML Markers download from spatial database. Once downloading is complete, an anonymous function manages XML file parsing and looping through all monitoring points. For each point, following actions are performed:
    1. geographic coordinates and code extraction;
    2. marker creation, by passing a marker options literal to constructor, including, among others, position, title (to be shown in a tooltip on mouseover event), icon, shadow, shape (clickable area within icon) and booleans pointing at draggable and flat properties;
    3. markers event listeners creation to manage mouse click, leading to info window visualisation (with a zoomed in map) and updating of selected markers array, and mouse over/out leading to 3D effects for improved user experience;
    4. marker addition to clustering management object.

Map extension is then reset to monitoring points minimum bounding rectangle, clustering management object inizialized and anonymous functions assigned to events associated with GUI elements.
Among them, time series button, providing asynchronous access to time series monitoring data of currently selected points. Dates and values are extracted from downloaded XML file and properly formatted within a data table , which is then passed to the dynamic charting widget.

On server side, PHP programs (app. 4) enable access to monitoring points spatial locations (tab. 3), measurement types (tab. 4) and time series data (tab. 5). They open the connection with the PostGIS spatial database, build a query by including parameters (if any) passed in from

the browser, issue the query, build a XML file using language DOM functions and send the file back to the browser.

```
- <markers>
    <marker code="m0002" lat="25.6102919095469" lon="59.4424922933168"/>
    <marker code="m0008" lat="25.5959449905844" lon="58.7614758097804"/>
    <marker code="m0016" lat="25.6788013181735" lon="58.5519825674601"/>
    <marker code="m0017" lat="25.6899817837729" lon="58.5586719670981"/>
    <marker code="m0020" lat="25.6738553982194" lon="58.6526098058984"/>
    <marker code="m0022" lat="25.5777950613951" lon="59.0686213210985"/>
    <marker code="m0028" lat="25.6487017974992" lon="59.1771043834618"/>
    <marker code="m0029" lat="25.6912835842241" lon="59.2343110031261"/>

        ...

    <marker code="m1319" lat="26.8817139715788" lon="56.0620216406374"/>
    <marker code="m1320" lat="26.9169897280828" lon="56.0560757474141"/>
    <marker code="m1321" lat="26.8962010737028" lon="56.0564405390827"/>
    <marker code="m1323" lat="26.9310690098161" lon="56.0410631703936"/>
    <marker code="m1342" lat="26.9252076767324" lon="56.03820526595"/>
  </markers>
```

Tab. 3 - Point locations XML dynamically generated file example

```
- <tstypes>
    <tstype id="1" code="h" variable="Piezometric head" units="m a.s.l." datatype="1" origin="1"/>
    <tstype id="2" code="v2" variable="v2" units="n.a." datatype="1" origin="1"/>
    <tstype id="3" code="v3" variable="v3" units="n.a." datatype="1" origin="1"/>
    <tstype id="4" code="v4" variable="v4" units="n.a." datatype="1" origin="1"/>
    <tstype id="5" code="v5" variable="v5" units="n.a." datatype="1" origin="1"/>
    <tstype id="6" code="v6" variable="v6" units="n.a." datatype="1" origin="1"/>
    <tstype id="7" code="v7" variable="v7" units="n.a." datatype="1" origin="1"/>
    <tstype id="8" code="v8" variable="v8" units="n.a." datatype="1" origin="1"/>
    <tstype id="9" code="v9" variable="v9" units="n.a." datatype="1" origin="1"/>
    <tstype id="10" code="v10" variable="v10" units="n.a." datatype="1" origin="1"/>
    <tstype id="11" code="v11" variable="v11" units="n.a." datatype="1" origin="1"/>
    <tstype id="12" code="v12" variable="v12" units="n.a." datatype="1" origin="1"/>
    <tstype id="13" code="v13" variable="v13" units="n.a." datatype="1" origin="1"/>
    <tstype id="14" code="v14" variable="v14" units="n.a." datatype="1" origin="1"/>
    <tstype id="15" code="v15" variable="v15" units="n.a." datatype="1" origin="1"/>
    <tstype id="16" code="v16" variable="v16" units="n.a." datatype="1" origin="1"/>
    <tstype id="17" code="v17" variable="v17" units="n.a." datatype="1" origin="1"/>
    <tstype id="18" code="v18" variable="v18" units="n.a." datatype="1" origin="1"/>
    <tstype id="19" code="v19" variable="v19" units="n.a." datatype="1" origin="1"/>
    <tstype id="20" code="v20" variable="v20" units="n.a." datatype="1" origin="1"/>
    <tstype id="21" code="v21" variable="v21" units="n.a." datatype="1" origin="1"/>
    <tstype id="22" code="v22" variable="v22" units="n.a." datatype="1" origin="1"/>
    <tstype id="23" code="v23" variable="v23" units="n.a." datatype="1" origin="1"/>
    <tstype id="24" code="v24" variable="v24" units="n.a." datatype="1" origin="1"/>
    <tstype id="25" code="v25" variable="v25" units="n.a." datatype="1" origin="1"/>
    <tstype id="26" code="v26" variable="v26" units="n.a." datatype="1" origin="1"/>
    <tstype id="27" code="v27" variable="v27" units="n.a." datatype="1" origin="1"/>
  </tstypes>
```

Tab. 4 – Measurement types dynamically generated XML file example

```
- <timeseries>
    <measure mdate="1997-09-06" value="11.41"/>
    <measure mdate="1997-10-09" value="11.58"/>
    <measure mdate="1997-11-07" value="11.4"/>
    <measure mdate="1997-12-03" value="11.4"/>
    <measure mdate="1998-01-08" value="11.46"/>
    <measure mdate="1998-02-03" value="11.41"/>
    <measure mdate="1998-03-03" value="11.42"/>
    <measure mdate="1998-04-04" value="11.51"/>
    <measure mdate="1998-05-06" value="11.5"/>
    <measure mdate="1998-06-02" value="11.48"/>
    <measure mdate="1998-07-03" value="11.47"/>
    <measure mdate="1998-08-05" value="11.5"/>
    <measure mdate="1998-09-04" value="11.41"/>
    <measure mdate="1998-10-02" value="11.4"/>
  </timeseries>
```

Tab. 5 – Time series dynamically generated XML file example

CASE STUDY

General framework

A groundwater monitoring case study, hypothetically located in south Iran, has been set up, in order to assess prototype adoption feasibility and its major potential benefits and bottlenecks in the framework of a realistic multi-year regional scale hydrogeological project.

Monitoring point locations and related time series are not consistent with local topographic and hydrogeological conditions (i.e. accessibility, elevation, aquifers and hydrochemical properties). However general framework, as groundwater flow directions resulting from piezometric heads regionalization[44] and salinization conditions, are potentially realistic, face to sea boundary located at southern limit of the study area. Data consistency issues have not been further investigated.

Data set size, namely number of point locations and related time series data respectively in the order of 1300 and 37000, effectively represents a realistic data set. A regional multi-year characterization hydrogeological project generally involves the design and implementation of a monitoring network and a monthly or quarterly monitoring plan, data adquisition frequency tipically depending upon variable type, its environmental sensitivity and time-space variability, as well as any further logistic, financial and contractual constraint.

The example would equally apply to typical local scale environmental characterization and remediation projects, which, despite the limited extent of investigation areas, generally impose high measurement density in the framework of long term monitoring addressing actions effectiveness assessment.

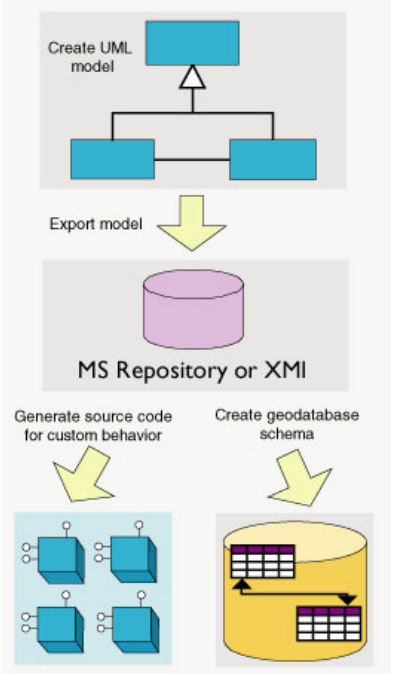
[44] Actually, piezometric interpretation rather than regionalization concept would be preferable, particularly in the context of hydrogeological analysis where relatively low measurement density generally leads to a strong bias towards interpretation and integration with other data sources in a multidisciplinary framework (Kresic, 2009)

Reference ArcGIS project setup

ESRI introduced, with ArcGIS 8, the concept of geodatabase, its own solution to spatial data management (MacDonald, 1999; Zeiler, 1999).

A geodatabase can be easily created in ArcCatalog, including its specific structures, as domains (user defined types at geodatabase level), feature datasets (a logical collection of features sharing the same spatial reference system), feature and object classes (to store geographic and attributes data, while implementing behaviours) and relationships (to model type inheritance, associations, aggregations, compositions, and their multiplicity).

An alternative more powerful approach points at database specialization of reference ArcInfo UML model. The original ArcInfo model, imported to a software engineering program as MsVisio, can be updated with new user-defined (feature and object) classes, inheriting from native high level classes (as spatial feature or object classes), and relationships. The extended model is then validated, exported to a MsAccess repository and later used as a blueprint for a geodatabase instance creation in ArcCatalog, as documented in ESRI (2001). The process is conceptually captured in **Fig. 22**.



http://edndoc.esri.com/arcobjects/9.1/ExtendingArcObjects/Appendices/UMLAndCASE.htm

Fig. 22 - ESRI Geodabase creation

Once created, spatial database objects can be dragged and dropped to ArcMap for visualisation, editing and analysis purposes. Provided that minimum licensing option[45] (namely ArcEditor level) is available, database constraints can be set up, enabling proper data integrity controls, further to supporting a more effective navigation experience in client GIS, as seamless access to time series data through simple mouse selection of geographic objects.

[45] http://www.gis.psu.edu/software/esrifaq.html

A realistic monitoring points data set has been initially structured in a proprietary ESRI ArcGIS 9.3 geodatabase, based on shifting and randomizing of original locations, in order to guarantee data privacy while still satisfying underlying hydrological spatial relationships. Spatial data set has been integrated with few polyline and polygonal features, schematically representing coastal line and subareas.
The database has then been populated with measurement types and time-dependent monitoring data.

Two snapshots (fig. 23) show the geodatabase architecture in ArcCatalog and a feature selection example through the ArcMap identify tool, enabling easy investigation of related time series. The composite relationship enables further integrity control, as cascade deleting, addressing automatic time series data removal when referenced monitoring points are deleted.

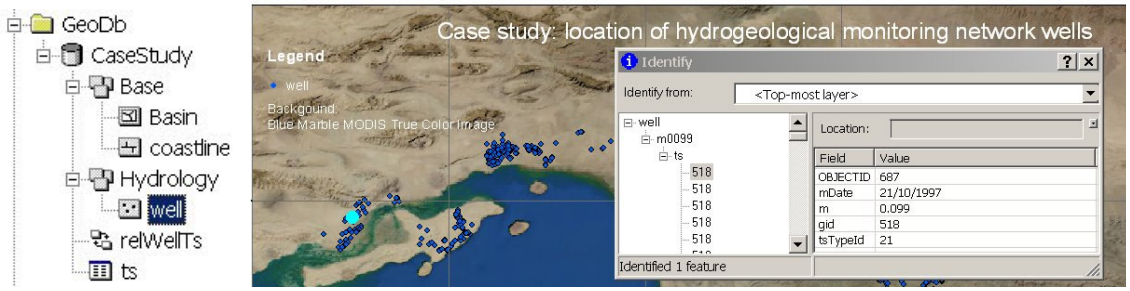


Fig. 23 – Snapshot of an ESRI geodatabase skeleton in ArcCatalog and of an editing session in ArcMap accessing data through a relationship class

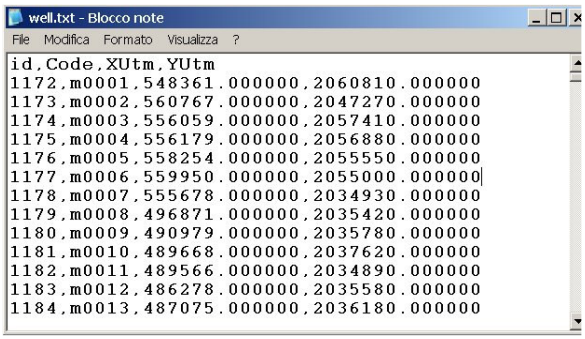
Previous chapters addressed issues related to compliance with OGC standards and possible impact of proprietary solutions, as the middle-tier ESRI ArcSDE, on later decisions to migrate to (or integrate) different database and GIS platforms, focusing on minimization of risks to be trapped in a "*pool of knowledge*" infrastructure. All these issues are part of the justification of major trends toward GIS open source technology adoption, as well as the increasing openness of proprietary platforms, and inherent motivation of current dissertation.

Data migration to PostGIS spatial database

PostGIS provides support to different approaches to massive external data loading, namely:

- a utility known as '*PostGIS Shapefile and DBF Loader*', accessible through pgAdmin III administrative tool, supporting direct ESRI shapefiles loading;
- the SQL COPY statement, acting on a text file stored on server side;
- the plsql \copy statement, acting on a text file stored on client side.

Operatively, data have been exported from the ESRI geodatabase to comma-delimited text files (fig. 24), respectively containing well locations and related time series data. PostGIS spatial database has been cleaned from previously stored testing data and a loading strategy, aimed at maintaining original geodatabase PKs and FKs, has been put in place following steps below:



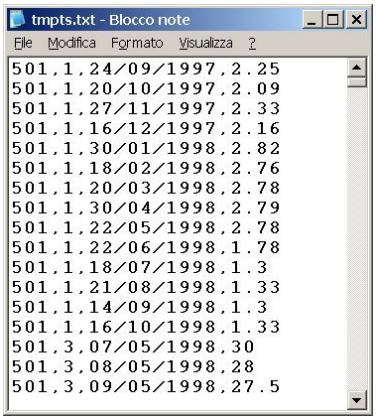


Fig. 24 – Text files exported from ESRI

- populate PostGIS spatial table g, based on a temporary intermediate table tmpWell storing original geodatabase id, code and UTM40N (SRID 32640) projected x and y coordinates. INSERT statement includes nested calls to PostGIS functions ST_MakePoint to create a point from distinct x and y coordinates, ST_SetSRID to set original UTM40N projection system and ST_TRANSFORM to convert to WGS84 geographic coordinates, consistently with original PostGIS specifications.
- populate tsType table, a straightforward task through an INSERT SQL statement.
- populate time series table, ts. As data attributes include, among others, measurement acquisition date, the preferred strategy has been to write an utility VBA program (app. 5), which, based on the original text file, builds a long SQL INSERT statement including calls to to_date PostgreSQL function.

It's worth noting that additional challenges are posed by multiple import sessions. When relying upon PostGIS automatic PK generation from a sequence, conflicts potentially arise with previously inserted records. A straightforward solution, not implemented at this stage, is to provide a simple trigger, automatically retrieving next sequence value in case of conflict.

PostGIS Spatial data delivery

Data stored to spatial database can be accessed in quite different ways.
PostgreSQL query builder (**Fig. 25**) provides the most powerful and flexible tool to query (as well as to build) the database, but the user must have authorizations, generally reserved to database developer, and have at least basic SQL knowledge.

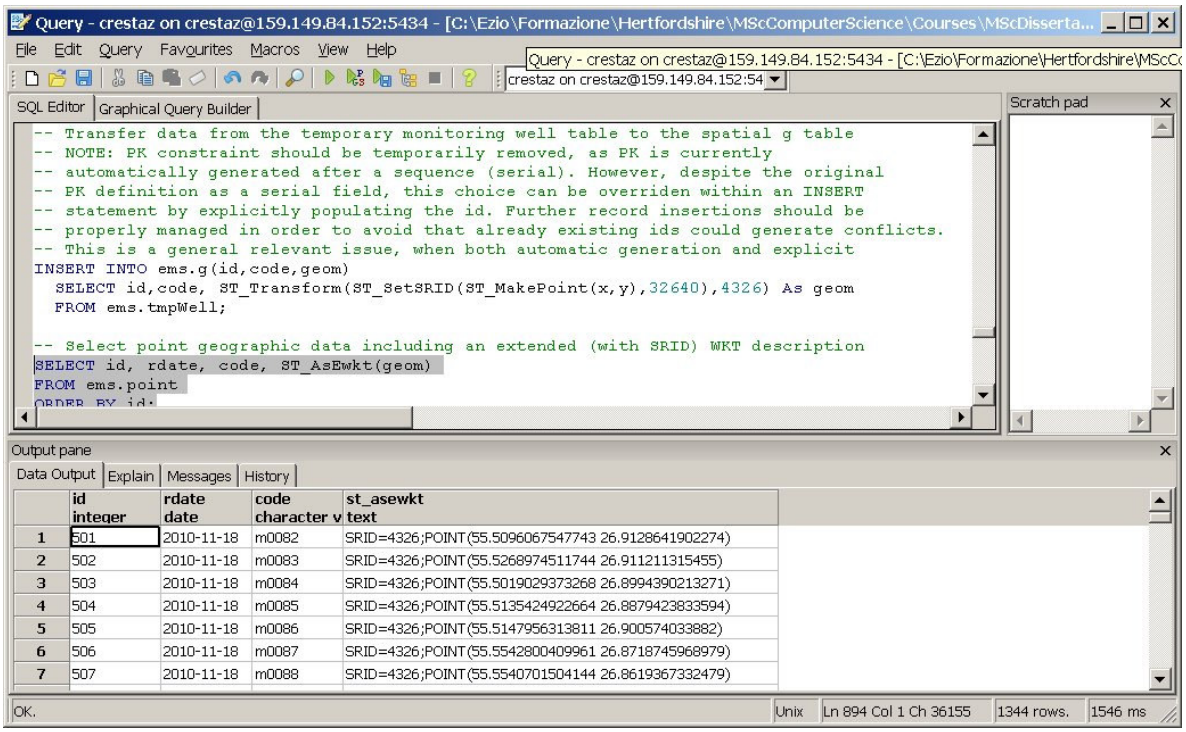


Fig. 25 – PostgreSQL/PostGIS query builder

Other users are expected to interact with the data source through dedicated desktop or web mapping/GIS applications.
Possibly the simplest approach is to directly connect from a Desktop GIS, as documented in QuantumGIS for a pointView database view (**Fig. 26**). Practical concerns in QuantumGIS

include that accessed spatial table/view must be registered in metadata PostGIS table *'geometry_columns'*, stored to public schema (**Fig. 27**), and that the spatial reference system must be explicitly set in QuantumGIS.

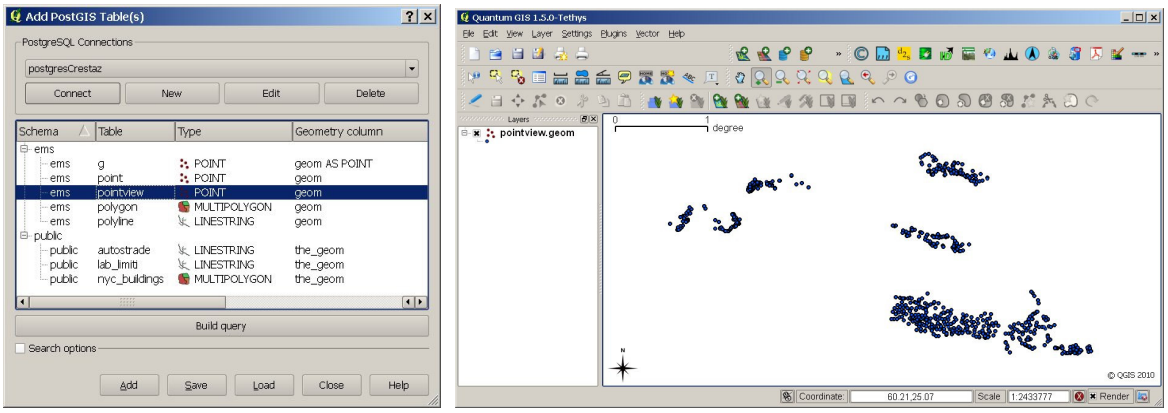


Fig. 26 – PostGIS data source access and visualisation through QuantumGIS

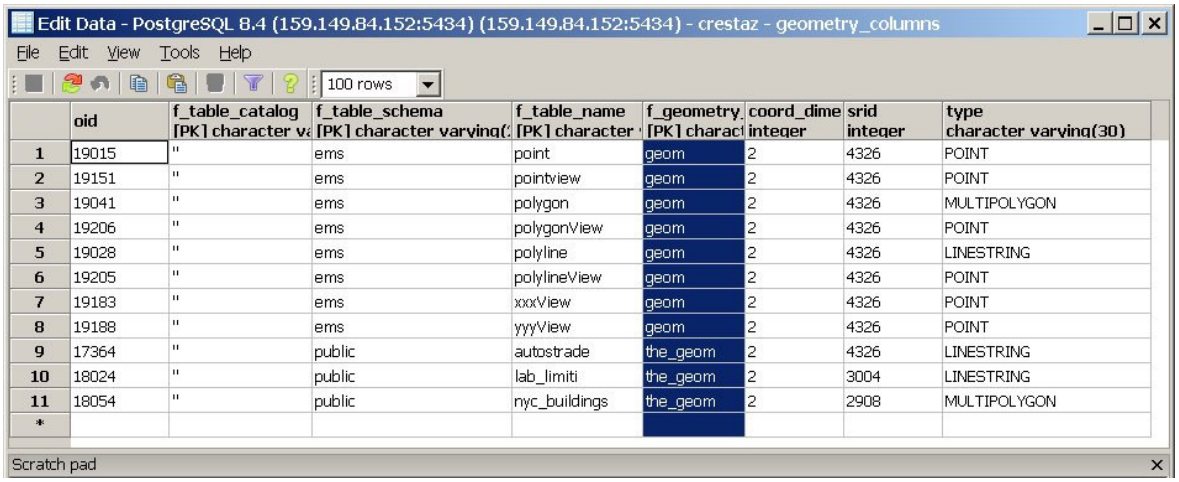


Fig. 27 - PostgreSQL/PostGIS spatial metadata table

Preliminary tests confirmed potential performance bottlenecks, resulting in long latency at database features loading or selection. The issue, involving – among others - assessment of communication bandwith, server and software fine tuning (Wiles, N.A.), should be further investigated before proceding on a real case study application.

Another delivery option is to rely upon a geographic server, as MapServer or GeoServer. The latter has been used in this research to test data delivery as OGC compliant WMS and WFS services, which can be consumed by specialized geographic clients, as OS Quantum GIS or proprietary ESRI ArcGIS (at the moment supporting only WMS standard), as well as within web mapping applications, i.e. built upon advanced java script geographic frameworks as GeoExt (2010) integrating ExtJs and OpenLayers. A fully worked example can be found at OpenGeo (2010).

Few snapshots of the Google Maps application in action (Fig. 28 and Fig. 29) document the user navigation experience, when accessing data both in space and time. Further to stressing the effectiveness of spatial clustering algorithms, snapshots point at the interactive user experience, based on points spatial selection and visualisation of related time series.

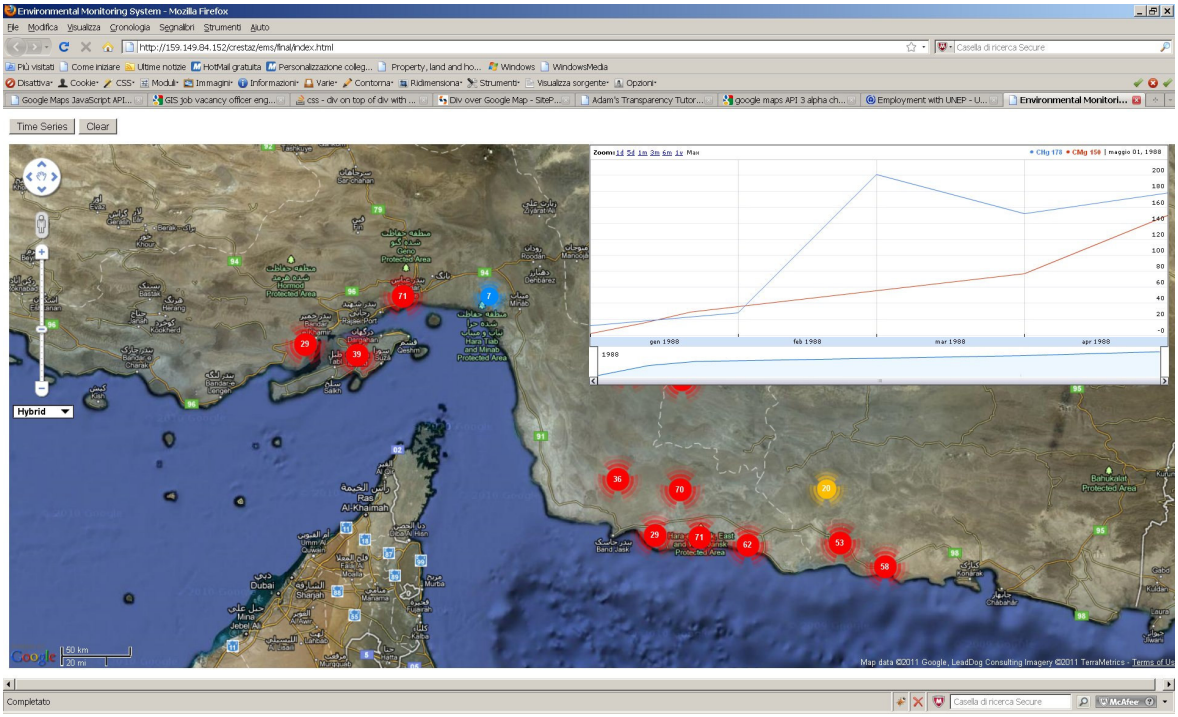


Fig. 28 – Hybrid map overview and monitoring points clustering in action

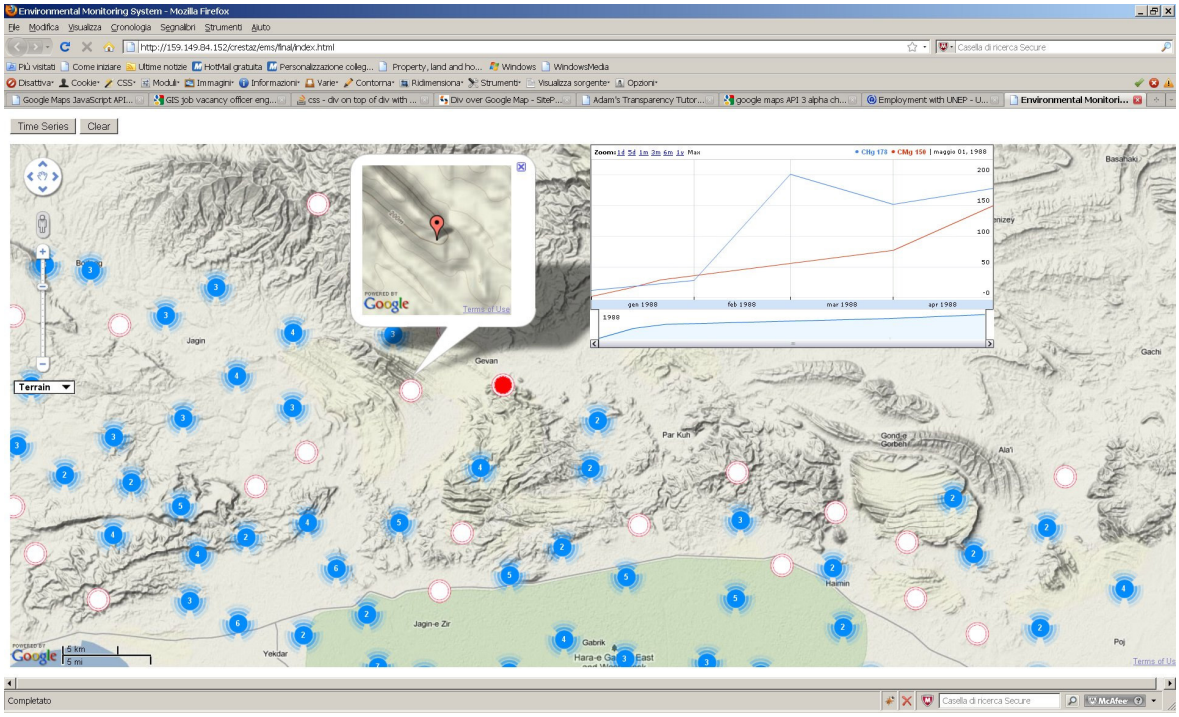


Fig. 29 – Time series access

CONCLUSIONS

Current research provided a brief review of open source advancements in geospatial industry and particularly in spatial databases, highlighting the complex relationships among most of ongoing projects and the high maturity level of few of them.

A gentle introduction to relational database theory foundations and to major impacts, benefits and shortcomings of object relational extensions, introduced the emergence of spatial database platforms, as Spatial Oracle/Locator, ESRI geodatabase and the OS PostgreSQL/PostGIS. The review pointed at their new specialized spatial data types, indexing methods, projection support, and, although at quite different levels, GIS vector and raster data models support.

The research then focused on the geodatabase templates, which have been designed and developed in the framework of joint collaboration efforts between ESRI and leading companies and research institutions. These templates, further to addressing requirements of such specific environmental and natural sciences domains as forestry, geology and hydrology in the framework of the proprietary ESRI ArcGIS platform, also provide general useful database design guidelines. The hydro data model, addressing surface hydrology issues, has been briefly reviewed from the wider perspective of spatio-temporal environmental data management.

Next steps addressed design and development of a PostGIS spatial database prototype, fully exploiting platform table inheritance features. Although less sophisticated than Spatial Oracle/Locator, yet PostGIS inheritance has been used to accommodate different geometries through a top parent class, each monitoring object being identified by a unique ID at database level. Provided that a measurement type table keeps trace of different variables and related information as measurement units (i.e. ℃, mg/l), the key (environmental) time series table enables storing of measurement date, value, as well as foreign keys referencing geometric environmental object and measurement type.

The analysis of geostack options provided review of different OS projects addressing access to spatio-temporal data stored to the PostGIS database, including Desktop GISs (as natively integrated Quantum GIS), geographic web servers, as Java WMS, WFS and WFS-T OGC-compliant GeoServer, java script libraries as OpenLayers, and more ambitious framework

GeoExt (integrating OpenLayers and ExtJs) for development of rich-content web mapping applications.

Provided some testing of spatial data delivery based on a geostack implementation based on above tools, the analysis of effectiveness of proposed spatio-temporal database solution has been addressed in the framework of a Google Maps API v. 3 web mapping application. Although the API is not OS, source code being not available, however it can be used for free with limited restrictions (as denial to remove Google logo and limited web traffic). The API provides a flexible and elegant OO implementation, while powerful Google spatio-temporal visualisation and analysis widgets enable implementation of relevant application features. Lessons learned at this stage are perfectly applicable also in OpenLayers.

The Google Maps application integrated with server side PHP scripting, further to documenting web AJAX programming techniques, highlights how easy remotely hosted spatial data can be accessed, although limited to point geometries for inherent performance bottlenecks. Particularly relevant from the HCI (Human Computer Interaction) point of view, the prototype addresses the key issue of effective accessibility to time series data based on environmental objects spatial selection. By providing the tools for measurement type selection, the application enables fast time series retrieving and visualization in time-dependent dynamic charts, based on points selected on cartography.

SQL, java script, PHP and Visual Basic utility source codes are fully documented in appendices.

A case study, focused on a realistic data set for a typical not automated monitoring project with a life span in the order of few years accounting more than 1000 locations and around 37000 time-dependent hydrogeological and hydrochemical measurements, has been set up, to investigate proposed solution effectiveness. No comprehensive test about application responsiveness under normal or stress conditions has been performed. However the adoption of clustering libraries and time visualisation widgets resulted very effective, with acceptable loading times despite the lack of optimization strategies on server side.

The conceptual nature of current research has the major advantage to focus on assessment of general design/development guidelines. Although mainly aimed at environmental management, the prototype could be refined and expanded to address the requirements of totally different

domains, which would potentially take advantage from a better integration of spatial and temporal dimensions, as Document Management Systems and stock analysis (Equis, 2002) in financial industry.

On the other hand, research limitis are inherently tied to the complexity of real case spatio-temporal management systems, which should address – among others – critical issues as:

- documents workflow management, particularly relevant in mega sites environmental assessment and remediation projects, which involve complex and interrelated technical, administrative and organizational procedures;
- legislation limits and visualization of spatial distribution and temporal trends of contamination
- higher integration of spatial, temporal and attribute dimensions, to effectively support spatio-temporal exploration (Roth and Ross, 2009) and thematic mapping (Sandvic, 2008[a,b]) in web applications.

The way forward

Based on current research, few short-term further steps have already been planned, namely:

- preparation of a paper to be submitted at next IGWMC at Colorado School of Mines sponsored conference "*MODFLOW and More 2011: Integrated Hydrologic* Modeling", to be held in Colorado on next 5-8 June. The conference includes a session dedicated to new developments in graphical user interfaces, visualization and GIS software, which is a particularly hot topic for hydrological integrated modelling and management systems design and development;
- updating of course notes and exercises of modules on advanced database design and environmental GIS, I have been teaching this year at Birkbeck College in London in the framework of current academic year MSc DL/ET in GISc.

In the long period, research outcomes will reasonably contribute to real case study applications and hopefully to better integration of GIS and environmental modelling in the mainstream IT, at least addressing previously stated limits and major organisational issues.

Software

Further to source code reported in appendices, two zip files are also provided:

- CrestazEzio_PostGISApplication_SourceCode.zip, which contains full SQL source code, as reported in app. 2; The source code can be used to recreate and test, also locally, the spatio-temporal database in PostGIS pgAdmin III query editor.
- CrestazEzio_GoogleMapsApplication_SourceCode.zip, which contains the Google Maps web mapping application. The application is ready to be transferred to a web server, provided that PostgreSQL and PostGIS are properly installed and populated, and atabase connection parameters properly set in PHP source code.

Google Maps application, providing access to spatial-temporal PostGIS database, can be accessed at following web address:
http://159.149.84.152/crestaz/ems/final/index.html

The application will be possibly soon moved to another server, due to maintenance activities. Updated information will be provided at:
http://www.giscience.it/en/index_en.html#Links

REFERENCES

Ambler S.W., N.A. UML 2 Class Diagrams. Available online: http://www.agilemodeling.com/artifacts/classDiagram.htm

Ambler S.W., 2004. *The Object Primer: Agile Model Development with UML 2*. 3rd Ed., Cambridge Press Un., UK

Anderson M.P. and Woessner W.W., 2002. *Applied Groundwater Modeling: Simulation of Flow and Advective Transport*. Academic Press, Elsevier, USA

Anselin L., Sridharan S. and Gholston S.,. 2007. Using Exploratory Spatial Data Analysis to Leverage Social Indicator Databases: The Discovery of Interesting Patterns. Social Indicators Research, 82(2): 287-309.

Arnold T. and Luinstra B., 2009. FEFLOW for science-based water protection: A watershed manager perspective. 2nd Feflow Conference, Postdam, Berlin, Germany

Avén N., 2010. PostGISonline web site: http://postgisonline.org/about.php? Last accessed: August 8, 2010

Bailey T.C. and Gatrell A.C., 1995. *Interactive Spatial Data Analysis*. Prentice Hall-Pearson Education Limited, UK

Batty M., Crooks A., Hudson-Smith A., Milton R., Anand S., Jackson M. and Morley J., 2010. Data mash-ups and the future of mapping. JISC, Technology & Standards Watch (TechWatch). Available online: http://www.jisc.ac.uk/media/documents/techwatch/jisctsw_10_01.pdf Last accessed: December 10, 2010

Bear J. and Verruijt A., 1986.Modeling Groundwater Flow and Pollution; Theory and Applications of Transport in Porous Media. Reidel Publishing Co., Dordrecht, Holland

BING, 2010. Web site: http://www.bing.com/maps Last accessed July 23, 2010

Bobrowski S., 2006. *Hands-On Oracle Database 10g Express Edition for Windows*. Osborne racle Presse Series

Britton C. and Doake J., 2005. *A student guide to object-oriented development*. Elsevier Butterworth-Heinemann, Oxford, UK

Brown M.C., 2006. *Hacking Google Maps and Google Earth*. Wiley Publishing Inc., Indianapolis, Indiana, USA

Burke R., 2003. *Getting to Know ArcObjects: Programming ArcGIS with VBA*. ESRI Press, Redlands, CA, USA

Burrough P.A. and McDonnell R.A., 1988. Principles of Geographical Information Systems. Oxford Press, Oxford, UK

Cascelli E., Crestaz E., Foglini F., Khalid F. and Power C., 2005. Geographical Information Systems and Groundwater Mathematical Modelling ArcGIS-FEFLOW Integration Case Study. Int. Conf. “GIS and Spatial Analysis” - IAMG 2005, Toronto, Canada, August 21-26, 2005

Chang K.T., 2005. *Programming ArcObjects with VBA: A Task-Oriented Approach*. CRC Press, FL, USA

Chen D., Carmona-Moreno C., Leone A. and Shams S., 2008. Assessment of Open Source GIS Software for Water Resources Management in Developing Countries. JRC Scientific and Technical Reports, EUR 23705 EN, JRC European Commission, IES Institute for Environment and Sustainability

Chmakov S., Hesch W., Tu C., Lima M. And Sichev P., 2009. Conceptual model development for FEFLOW or MODFLOW models – a new generation of Schlumberger Water Services software. 2nd Feflow Conference, Postdam, Berlin, Germany

Chow T.E., 2008. The Potential of Maps APIs for Internet GIS Applications.. Transactions in GIS, 12(2), pp. 179-191

Codd E.F., 1970. A Relational Model of Data for Large Shared Data Banks. Comm. ACM, 13(6), 377-387

Connolly T. and Begg C., 2010. *Database Systems: A practical Approach to Design, Implementation, and Management*. 5th Ed., Addison Wesley

Conway J.E., 2009. PL/R User's Guide – R Procedural Language. Available online at: http://www.joeconway.com/plr/doc/index.html

Conway J.E., 2010. Web site: http://www.joeconway.com/web/guest

Crestaz, 2003. Groundwater Geographical Information Systems – Investigation of a Coupled Data Management and Numerical Flow Modelling strategy for ArcGIS 8. Unpublished Msc thesis, Unigis - Manchester Metropolitan University, Uk

CRWR, 2010. Center for Research in Water Resources web site: http://www.crwr.utexas.edu/ Texas University at Austin, USA Last accessed: November 14, 2010.

Daoyi C., Carmona-Moreno C., Leone A. and Shams S., 2008. Assessment of Open Source GIS Software for Water Resources Management in Developing Countries. JRC Scientific and Technical Reports. EUR 23705 EN-2008

De Smith M., Goodchild M.F. and Longley P.A., 2009. Geospatial Analysis: A Comprehensive Guide to Principles, Techniques and Software Tools. 3rd Ed., Leichester, Troubador and www.spatialanalysisonline.com

DHI-WASY, 2010. FEFLOW groundwater flow and transport modelling system. Web site: http://www.feflow.info/

Diersch H.J.G., 2005. *FEFLOW, Finite Element Subsurface Flow & Transport Simulation System*, White Papers Vol. I. Wasy, Institute for Water Resources Planning and Systems Research Ltd., Berlin, Germany

Dosi C. Eds., 2001. Agricultural Use of Groundwater. Kluwer Academic Publishers

Dunfey R.I., Gittings B.M. and Batcheller J.K., 2006. Towards an open architecture for vector GIS. Computer & Geosciences 32 (2006), pp. 1720-1732

EPSG, 2010. Web site: http://www.epsg.org/ Last accessed: August 17, 2010

Equis, 2002. Metastock Professional for Windows 98 and higher. User's Manual, Version 8.0. Salt Lake City, Utah, USA

ESRI, 1997. Understanding GIS: The ArcInfo Method. ESRI Press, Redlands, CA

ESRI, 1998. ESRI Shapefile Technical Description. ESRI White Paper, July 1998. Available online at: http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

ESRI, 1999. *Understanding ArcSDE*. Available online at: http://www.zmuc.dk/public/GIS-intro/Litteratur/ESRIdigitalBooks/Understanding_ArcSDE.pdf

ESRI, 2001. CASE Tools Tutorial: Creating custom features and geodatabase schemas. Available at: http://edndoc.esri.com/arcobjects/8.3/ (Samples > Case Tools > Custom Features)

ESRI, 2003. ArcGIS: Working With Geodatabase Topology. ESRI White Paper, May 2003. Available online from: http://www.esri.com/library/whitepapers/pdfs/geodatabase-topology.pdf

ESRI, 2005. Arc Hydro – HydroID. Version 1.1 Final, July 2005. Available online at: http://www.crwr.utexas.edu/gis/gishydro06/ArcHydro/ArcHydroTools/Doc/Unique%20ID%20Manager%20-%20User%20Manual.pdf

ESRI, 2010. Data models. Available online at: http://resources.arcgis.com/content/data-models. Last accessed: September 17, 2010

ESRI, 2011. Company web site: http://www.esri.com/ Last accessed January 23, 2010

ESTAT, 2010. Web site: http://www.geovista.psu.edu/ESTAT/ Last accessed July 23, 2010

EPA, 2009. WATERS Web, Mapping, and Database Services. Available at:

http://www.epa.gov/waters/geoservices/index.html Last update: September 29th, 2009

eVis, 2010. The Event Visualization Tool Web site: http://biodiversityinformatics.amnh.org/open_source/evis/index.php Last accessed: August 11, 2010

Fehily C., 2008. SQL, 3rd Ed., Peachpit Press

Forer P., 1998. Geometric Approaches to the Nexus of Time, Space, and Microprocess: Implementing a Practical Model for Mundane Socio-Spatial Systems. In Egenhofer M.J. and Golledge R.G. (eds.), 1998. *Spatial and Temporal Reasoning in Geographic Information Systems*, Oxford Un. Press, New York, pp. 171-190

Frank A.U., 1998. Different Types of Time in GIS. In Egenhofer M.J. and Golledge R.G. (eds.), 1998. *Spatial and Temporal Reasoning in Geographic Information Systems*, Oxford Un. Press, New York, pp. 40-62

FGDC, 2010. Federal Geographic Data Committee web site: http://www.fgdc.gov/metadata Last accessed: October 3, 2010

FSF, 2010. Free Software Foundation web site: http://www.fsf.org/ Last accessed July 18, 2010

Gapminder, 2011. Web site: http://www.gapminder.org/ Last accessed: January 24, 2011

GDAL/OGR, 2010. Geospatial Data Abstraction Library web site: http://www.gdal.org/ Last accessed July 23, 2010

GeoDA, 2010. Web site: http://geodacenter.asu.edu/. Last accessed: July 23, 2010

GeoExt, 2010. Web site: http://www.geoext.org/ Last accessed: September 19, 2010

GeoServer, 2010. Web site: http://geoserver.org/display/GEOS/Welcome. Last accessed: July 10, 2010

GeoTools, 2010. Web site: http://docs.codehaus.org/display/GEOTOOLS/Home Last accessed July 23, 2010

Gocu R.C., Carabin G. Hallet V., Peters V. and Dassargues A, 2001. GIS-based hydrogeological database and groundwater modelling. Hydrogeology Journal 9, pp. 555-569

Goofchild M.F., 2007. Citizens as sensors: web 2.0 and the volunteering of geographic information. GeoFocus (Deitorial), n. 7, p. 8-10

Google, 2010. Google Maps JavaScript API V3 Web site: http://code.google.com/intl/it/apis/maps/documentation/javascript/ Last accessed: December 1, 2010

GRASS, 2010. Web site: http://grass.itc.it/ Last accessed: July 23, 2010

gvSIG, 2010. Web site: http://www.gvsig.org/web/ Last accessed: July 23, 2010

Hazelton N.W.J., 1998. Some operational Requirements for a Multi-Temporal 4-DGIS. In Egenhofer M.J. and Golledge R.G. (eds.), 1998. *Spatial and Temporal Reasoning in Geographic Information Systems*, Oxford Un. Press, New York, pp. 63-73

Hengl T., 2009. A Practical Guide to Geostatistical Mapping. Available online at: http://spatial-analyst.net/book/

Hogeweg M., 2000. *Spatio-temporal visualisation and analysis*. MSc dissertation, Salford Un., UK. May 2000. Unpublished

Karanjac J., 1993. United Nations Ground Water Software. Ground Water, vol. 31, issue 2, pp. 311-315, 4 March 1993

Kothuri R., Godfrind A. and Beinat E., 2004. *Pro Oracle Spatial: The essential guide to developing spatially enabled business applications*. APress, Berkeley, CA, USA

Kraak M.J., 2005. Visualising spatial distributions. In Longley PA., Goodchild M.F., Maguire D.J. and Rhind D.W. (eds.), 2005. *Geographical Information Systems: Principles, Techniques, Management and Applications*. 2nd Ed., John Wiley & Sons, NJ, USA, pp. 49-65

Kresic, N., 2009. *Groundwater Resources: Sustainability, Management, and Restoration*. The McGraw-Hill Companies, Inc.

Kresse W. and Fadaie K., 2004. ISO Standards for Geographic Information. Springer-Verlag, Berlin, Germany

Kropla B., 2005. *Beginning MapServer: Open Source GIS Development – Create spatially-enabled Web application using the MapServer GIS development environment*. Apress, Berkeley, CA, USA

Igualada F.J., 2009. February 2009. Entrevista. Cuadernos Internacionales de Tecnologia para el Desarollo Humano.- Tecnologia de Informaciòn Geografica. Available online at: http://www.cuadernos.tpdh.org/file_upload/08_TIG_11_igualada.pdf Last accessed: October 15, 2010

ILWIS, 2010: Web site: http://www.ilwis.org/ Last accessed July 23, 2010

ISO, 2010. International Standard Organization web site: http://www.iso.org/iso/home.html Last accessed July 23, 2010

Jankowski P., Tsou M.H. and Wright R.D., 2007. Applying Internet Geographic Information System for Water Quality Monitoring. Geography Compass 1/6 (2007): 1315-1337, Blackwell Publishing Ltd.

JTS, 2010. Java Topology Suite web site: http://www.vividsolutions.com/jts/jtshome.htm Last accessed July 23, 2010

Lake R., Burggraf D., Trninic M. and Rae L., 2004. *Geography Mark-Up Language: Foundation for the Geo-Web*. John Wiley & Sons, Chichester, UK

Langran G., 1992. Time in Geographic Information Systems. Taylor & Francis, London, UK

Lillesand T.M. and Kiefer R.W., 2000. *Remote Sensing and Image Interpretation*. 4th Ed., John Wiley & Sons, USA

Longley PA., Goodchild M.F., Maguire D.J. and Rhind D.W., 2005[a]. Geographic Information Systems and Science. 2nd Ed., John Wiley and Sons, NJ, USA

Longley PA., Goodchild M.F., Maguire D.J. and Rhind D.W. (eds.), 2005[b]. *Geographical Information Systems: Principles, Techniques, Management and Applications*. 2nd Ed., John Wiley & Sons, NJ, USA

Maasdam R. 2000. *Exploratory Data Analysis in Water Quality Monitoring Systems*. Unpublished MSc thesis at Salford Un., UK Available online at: http://www.feweb.vu.nl/unigis/downloads/msc/Ruurd%20Maasdam.pdf

MacDonald A., 1999. *Building a geodatabase*. ESRI Press, Redlands, CA, USA.

Maidment D.R., 2002. Arc Hydro: GIS for Water Resources. ESRI Press, Redlands, CA, USA

Maguire D.J., 2005. GIS customisation. In Longley PA., Goodchild M.F., Maguire D.J. and Rhind D.W. (eds.), 2005. *Geographical Information Systems: Principles, Techniques, Management and Applications*. 2nd Ed., John Wiley & Sons, NJ, USA, pp. 149-159

Mapserver, 2010. Web site: http://mapserver.org/ Last accessed July 23, 2010

MapWindow GIS, 2010. Web site: http://www.mapwindow.org/ Last accessed July 23, 2010

Marey E.J., 1885. La Méthode Graphique. Paris, France

Masser I., 2005. *GIS Worlds: Creating Spatial Data Infrastructures*. ESRI Press, Redlands, CA, USA

Micklin P., 2007. *The Aral Sea Disaster*. Annu. Rev. Earth Planet. Sci. 2007

NASA, 2010. Jet Propulsion Laboratory at California Institute of Technology web site: http://onearth.jpl.nasa.gov/ Last accessed September 18, 2010

Neteler M. and Raghavan V., 2006. Advances in Free Software Geographic Information Systems. Journal of Informatics, 3(2)

Obe R.O. and Hsu L.S., 2010. *PostGIS in Action*. Manning Publications Co. Unedited Draft, last update: 27/5/2010. Available for purchase from: http://www.manning.com/obe/

OGC, 2010. Open GIS Consortium wbe site: http://www.opengeospatial.org/ Last accessed July 23, 2010

OpenGeo, 2010. Web site: http://workshops.opengeo.org/postgis-spatialdbtips/#getting-started Last accessed: September 6, 2010

OpenJump, 2010: Web site: http://www.openjump.org/ Last accessed July 23, 2010

OpenLayers, 2010. Web site: http://openlayers.org/ Last accessed July 23, 2010

Oracle, 2010. Company web site: http://www.oracle.com/index.html Last accessed July 23, 2010

O'Relly T., 2005. What is Web 2.0. Design Patterns and Business Models for the Next Generation of Software. Web site: http://oreilly.com/pub/a/web2/archive/what-is-web-20.html?page=1 Last accessed: January 24, 2011

OSI, 2010. Open Source Initiative web site: http://www.opensource.org/ Last accessed July 18, 2010

Ott T. and Swiaczny F., 2001. *Time-integrative Geographic Information Systems – Management and Analysis of Spatio-Temporal Data*. Springer Verlag, Berlin

Peng Z.R. and Tsou M.H., 2003. *Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks*. John Wiley & Sons, NJ, USA

Perens P., 1999, The Open Source Definition. In: Dibona C., Ockman S. and Stone M., eds. Open Sources: Voices from the Open Source Revolution, Cambridge, MA: O'Reilly, 171–188. Available from: http://perens.com/Articles/OSD.html Last accessed July 23, 2008

Peterson M., n.a.. Online Mapping with Google Maps API v3. Available at: http://maps.unomaha.edu/workshops/GoogleMapsCode/Online_Mapping_with_the_Google_Maps_API.pdf Last accessed December 10, 2010

PgRouting, 2010. Web site: http://www.pgrouting.org/ Last accessed December 1, 2010

PL/R, 2010. R Procedural Language for PostgreSQL web site: http://www.joeconway.com/plr/ Last accessed December 1, 2010

PostGIS, 2010. Web site: http://postgis.refractions.net/ Last accessed July 23, 2010

PostGIS Raster, 2010. Web site: http://trac.osgeo.org/postgis/wiki/WKTRaster. Last accessed December 1, 2010

PostgreSQL, 2010. Web site: http://www.postgresql.org/ Last accessed July 18, 2010

Proj.4, 2010. Web site: http://trac.osgeo.org/proj/ Last accessed July 23, 2010

Purvis M. , Sambells J. and Turner C., 2006. *Beginning Google Maps Applications with PHP and AJAX: From Novice to Professional: Build awesome web-based mapping applications with this powerful API* APress, Berkeley, CA, USA

Quantum GIS, 2010. Web site: http://www.qgis.org/ Last accessed July 23, 2010

Racine P., 2010. PostGIS WKT Raster Tutorial 1 – Intersecting vector polygons with large raster coverage using PostGIS WKT Raster. June 2010., Available online at: http://trac.osgeo.org/postgis/wiki/WKTRasterTutorial01

Ramsey, 2007. Introduction to PostGIS: Installation – Tutorial – Exercises. Available online at http://2007.foss4g.org/workshops/W-04/PostGIS%20Workshop.doc Last accessed: August 27, 2010

Raymond E.S., 2000. The Cathedral and the Bazaar. Available online at: http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/

Rigaux P., School M. and Voisard A., 2002. Spatial Databases: With Application to GIS (The Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers, Elsevier, San Francisco, CA

Robinson A. C. 2005[a]. *Assessing Geovisualization in Epidemiology: A Design Framework for an Exploratory Toolkit*. Master's Thesis, The Pennsylvania State University Available online: http://www.geovista.psu.edu/ESTAT/estat_writing.html Last accessed: August 11, 2010

Robinson A.C., 2005[b]. Getting to Know ESTAT: The Exploratory Spatio-Temporal Analysis Toolkit. GeoVISTA Center, Dpt. Of Geography, Pennsylvania State Un., May 31, 2005. Available online: http://www.geovista.psu.edu/ESTAT/estat_practice.html Last accessed: August 11, 2010

Robinson A.C., Chen J., Lengerich E., Meyer H.and MacEachren A.M.. 2005[a]. Combining Usability Techniques to Design Geovisualization Tools for Epidemiology. Proceedings of Auto-Carto 2005. Las Vegas, NV, March 18-23, 2005 Available online: http://www.geovista.psu.edu/ESTAT/estat_writing.html Last accessed: August 11, 2010

Robinson A.C. 2005[b]. Geovisualization and Epidemiology: A General Design Framework. Proceedings of the International Cartographic Association, La Coruna, Spain, July 9-16 Available online: http://www.geovista.psu.edu/ESTAT/estat_writing.html Last accessed: August 11, 2010

Roth R.E. and Ross K.S., 2009. Extending the Google Maps API for Event Animation Mashups. Cartographic Perspective, Number 54, Fall 2009, Available online: http://www.geovista.psu.edu/publications/2009/RothRoss_2009_CP.pdf Last accessed: December 2, 2010

SAGA, 2011. System for Automated Geographic Analysis web site: http://www.saga-gis.org/ Last accessed: January 24, 2011

Sandvic B., 2008. Using KML for Thematic Mapping. MSc in Geographical Information Science.dissertation. School of Geosciences, Un. of Edinburgh. Available online at: http://thematicmapping.org/downloads/Using_KML_for_Thematic_Mapping.pdf Last accessed: December 10, 2010

Sandvic B., 2008. Part 2: Supporting document - Thematic Mapping Engine. MSc in Geographical Information Science.dissertation. School of Geosciences, Un. of Edinburgh. Available online at: http://thematicmapping.org/downloads/Thematic_Mapping_Engine.pdf Last accessed: December 10, 2010

Sanz-Salinas J.G. and Montesinos-Lajara M., 2009. Current Panorama of the FOSS4G Ecosystem. The European Journal for the Informatics Professional. Vol. X, Issue No. 2, April 2009, pp. 43-51

Sencha Inc., 2010. Ext Cross-browser JavaScript library for rich web apps. Web site: http://www.sencha.com/products/js/ Last accessed: September 19, 2010

Steiniger S. and Bocher E., 2009. An Overview on Current Free and Open Source Desktop GIS Developments. International Jounal of Geographical Information Science. Vol. 23

Steiniger S. and Hay G.J., 2009. Free and Open Source Geographic Information Tools for Landscape Ecology. Ecological Informatics 4 (2009), p. 183-195, Elsevier

Stewart J., Tonini A., Igualada F. and Mugambi F., 2009. Terrain Analysis Support: Mobility Modeling for Peace-keeping Operations. ESRI Conference 2009 Proceedings, San Diego, CA, USA. Available online: http://proceedings.esri.com/dvd/uc/2009/uc/papers/pap_1767.pdf Last accessed: October 15, 2010

Strassberg G. and Maidment D., 2004. Arc Hydro groundwater data model. Available from: http://twri.tamu.edu/usgs/2003-04/strassberg_awra.pdf

Svennerberg G., 2010. *Beginning Google Maps API 3*. Apress, Berkeley, CA, USA

Tecplot, 2011. Web site: http://www.tecplot.com/ Last accessed: January 27, 2011

Tufte E., 1997. Visual Explanations: Images and Quantities, Evidence and Narrative. Graphics Press LLC. Cheshire, Connecticut, USA

Tufte E., 2001. *The Visual Display of Quantitative Information*. 2nd Ed., Graphics Press LLC. Cheshire, Connecticut, USA

Turton I., 2010. PennState Un. Open Web Mapping web site: https://courseware.e-education.psu.edu/courses/geog585/content/home.html Last accessed: August 8, 2010

uDig, 2010. Web site: http://udig.refractions.net/ Last accessed: July 23, 2010

UNEP, 2011. World Conservation Monitoring Center web site: http://www.unep-wcmc.org/ Last accessed: January 23, 2011

UNESCO, 1983. International legend for hydrogeological maps. Revised ed. 1983. Available from: http://unesdoc.unesco.org/images/0015/001584/158459eo.pdf

USGS, 2010. Glovis web site: http://glovis.usgs.gov/ Last accessed: July 23, 2010

Ushahidi, 2010. Web site: http://www.ushahidi.com/ Last accessed: December 1, 2010

van Deursen W.P.A., 1995. *Geographical Information Systems and Dynamic Models*, Ph.D. thesis, Utrecht University, NGS Publication 190, 198 pp. Available from: http://www.geog.uu.nl/pcraster/thesisWvanDeursen.pdf

Voigt R., 1998. Efficient handling of time-varying FEM model data in the 3D groundwater simulation system FEFLOW. 3rd Congress Graphik-gestutzie Grundwassermodellierung, 27-28 May 1998, Berlin, Germany

Weaver S.D., Bate M.C. and Autio R.J., 2003. *EquIS for ArcGIS: State-of-the-art environmental data management and analysis*. Available from: http://www.earthsoft.com/

Wiles F., N.A. Performance Tuning PostgreSQL. Available at: http://www.revsys.com/writings/postgresql-performance.html Last accessed: January 24, 2011

Williams M., 2010. Web site: Google Maps API Tutorial web site: http://www.econym.demon.co.uk/ Last accessed: December 11, 2010

Wood J., 2002. *Java Programming for Spatial Sciences*. Taylor & Francis, London UK

Worboys M.F., 1995. *GIS: A Computing Perspective*. Taylor and Fancis, London, UK

Worboys M.F., 1998. A Generic Model for Spatio-Bitemporal Geographic Information. In Egenhofer M.J. and Golledge R.G. (eds.), 1998. *Spatial and Temporal Reasoning in Geographic Information Systems*, Oxford Un. Press, New York, pp. 25-39

Worboys M.F., 2005. Relational databases and beyond. In Longley PA., Goodchild M.F., Maguire D.J. and Rhind D.W. (eds.), 2005. *Geographical Information Systems: Principles, Techniques, Management and Applications*. 2nd Ed., John Wiley & Sons, NJ, USA, pp. 163-174

Worboys M.F. and Duckman M., 2004. GIS: A Computing Perspective. 2nd Ed., Boca Raton, CRC Press

Yahoo, 2010. Yahoo! Maps Web Services: http://developer.yahoo.com/maps/ Last accessed: December 1, 2010

Zeiler, M.,1999. *Modeling our world*. ESRI Press, Redlands, CA, USA

Zeiler, M., 2001[a]. *Exploring ArcObjects: Vol. 1 – Applications and Cartography*. ESRI Press, Redlands, CA, USA

Zeiler, M., 2001[b]. *Exploring ArcObjects: Vol. 2 – Geographic Data Management*. ESRI Press, Redlands, CA, USA

APPENDIX 1 – RELATIONAL MODEL FOUNDATION CONCEPTS

Relational data model (Codd, 1970; Connolly and Begg, 2010: pp. 89-258), based on mathematical set theory, provides a simple and effective approach to data management.

Data are organized in tables, each row identifying an entity and each column representing a specific property after a given domain (a data type as text, date, number, plus any further constraint as acceptable values and nullability). Column entries must be atomic values (that's they can not be further subdivided) of the specified domain or nulls (if acceptable), provided that the 'atomic' concept is inevitably a relative granularity-dependent concept.

Each row is uniquely identified by a primary key (PK), simple or composite (one or multiple columns key). When several unique candidates are available, one of them is selected as PK, remaining options being known as alternate keys.
Foreign keys (FKs) are a column or group of columns in a table, whose value(s) refer to values (actually PK or any candidate key) in another table (or in the same table, when self-referencing). FKs provide the powerful referential integrity mechanism, to relate data splitted through different tables. FKs help enforcing referential rules, as cascade deleting which addresses the requirement, once a record is deleted, to automatically remove related records in related tables.

Common and equivalent terminology in relational model, SQL and files management is reported in table.

Model	SQL	Files
Relation	Table	File
Attribute	Column	Field
Tuple	Row	Record

Fehily (2008)
Terminology in relational model, SQL and files management

Relationships are association between columns in different tables and they can be of three distinct types: one-to-one (1:1), one-to-many (1:M) and many-to-many (M:M).
A straightforward example of the latter is the relationship between a spatial parcel table and a proprietary table. Each parcel can have more than one proprietary and, on the other hand, each proprietary can own more than one parcel. The relationship can be modelled by an intermediate table whose composite PK combines at least the two FKs pointing at parcel and proprietary

tables, while other data could converge to this intermediate table as well, as dates, addressing the requirements of an historical cadastral system, and propriety percentage.

Relational database can be created, queried and managed through SQL, which is a complete and simple declarative programming language.
SQL is (relationally) complete because it supports creation, deletion and use of databases, although still lacking of procedural programming constructs, which are supported in the framework of language extensions as PL_SQL in Oracle or psql in PostGIS.
It is simple as, through its limited number of pseudo-english statements, it supports both non expert and advanced users, in their day-by-day tasks or in performing complex operations.
It is declarative because, differently from 3rd generation languages as C++ or JAVA, it enables to express what you want, not how the task should be performed.

SQL statements can be conceptually organized in three distinct groups, namely:

- Data Definition Language (DDL), as create table, alter table, create view, create index and drop table/view/index;
- Data Manipulation Language (DML), as insert into, update, delete and select;
- Data Control Language (DCL), as grant and revoke.

Despite their relevance to most human data management activities, relational databases have for long time suffered of major performance bottlenecks, strongly limiting their adoption in application domains addressing complex data types. As previously stated, only recently the traditional dual architectures adopted in early days geospatial industry have been abandoned in favour of a true spatial database paradigm.

Further to increasing hardware and software overall performance, object extensions to traditional relational model played a key role in the development of new spatial database paradigms.

APPENDIX 2 – POSTGIS SPATIO-TEMPORAL DATABASE: SQL SOURCE CODE

SQL source code

```
-- ********************************************************************************************
-- Author: Ezio Crestaz
-- Date:   January 2011
-- Scope:  Creation of an environmental monitoring spatio-temporal  database (hereafter referred to with
--         the acronym ems). Source code is fully documented here below, including data insertion and
--         and deletion tests, and any further note relevant to future implementation issues.
--         The database model has been designed after the ESRI-CRWR (www.crwr.utexas.edu) Hydro data
--         model. Full references in Maidment D.R., 2002. Arc Hydro: GIS for Water Resources. ESRI Press
--         The database model has been tailored for PostGIS in order to make use of its table inheritance
--         features.
-- ********************************************************************************************

-- Check PostgreSQL and PostGIS versions
SELECT version();
SELECT postgis_full_version();

CREATE SCHEMA ems
  AUTHORIZATION crestaz;
COMMENT ON SCHEMA ems IS 'Environmental Management System
Beta release to investigate data modelling issues for effective environmental monitoring data management.';

-- Creation of geographic data tables
--
-- Parent spatial table for geographical features
-- Note: geometry field is not added at this stage, as it would not be possible to enforce the geometry
-- type constraints on child tables
-- Unfortunately a unique constraint on code can not be imposed, if we want to keep the model general,
-- accomodating data from different research centers, companies and geographical areas, possibly
-- sharing the same code.
-- Also data selection based on code will require to be integrated with further criteria, as geographical
-- ones (i.e. monitoring point code, i.e. 'Mp01', plus a proximity or topological containmente
-- geographical criteria, within a given region or closest to a given point). As the idea is to provide
--  a general purpouse platform, the issue of projections is relevant. Each spatial table must
```

```
34 -- be registered, including its spatial reference system, so the initial assumption of an undefined
35 -- SRID (Spatial Reference ID equal to -1) is not acceptable, nor it would be to limit data entry within
36 -- a specific geographic region and/or a given planar projection. For this reason, it has been decided
37 -- to assume that all data locations are entered as geographic coordinates in WGS84 (lat/lon with a
38 -- SRID=4326), which is by the way the typical reference system for GPS data (refer to AddGeometryColumn
39 -- function calls for further details). Existing projected data can be easily converted to WGS84, by
40 -- using the ST_Convert PostGIS function (more later)
41 CREATE TABLE ems.g(
42   id serial PRIMARY KEY,
43   rDate date NOT NULL,              -- Database recording date
44   code varchar(150)
45   );
46
47 -- Child table for point features
48 CREATE TABLE ems.point(
49   CONSTRAINT pk_point PRIMARY KEY (id))
50   INHERITS ("ems".g);
51
52 SELECT AddGeometryColumn ('ems','point','geom',4326,'POINT',2);
53
54 -- Child table for polyline features
55 CREATE TABLE ems.polyline(
56   CONSTRAINT pk_polyline PRIMARY KEY (id))
57   INHERITS (ems.g);
58
59 SELECT AddGeometryColumn ('ems','polyline','geom',4326,'LINESTRING',2);
60
61 -- Child table for polygon features
62 CREATE TABLE ems.polygon(
63   CONSTRAINT pk_polygon PRIMARY KEY (id))
64   INHERITS (ems.g);
65
66 SELECT AddGeometryColumn ('ems','polygon','geom',4326,'MULTIPOLYGON',2);
67
68 -- Create spatial indexes on child spatial tables
69 CREATE INDEX idx_point_geom ON ems.point USING GIST(geom);
70 CREATE INDEX idx_polyline_geom ON ems.polyline USING GIST(geom);
71 CREATE INDEX idx_polygon_geom ON ems.polygon USING GIST(geom);
72
73 -- Create btree indexes on code for each child table
74 CREATE INDEX idx_point_uCode
75 ON ems.point USING btree(upper(code));
```

```
 76
 77 CREATE INDEX idx_polyline_uCode
 78 ON ems.polyline USING btree(upper(code));
 79
 80 CREATE INDEX idx_polygon_uCode
 81 ON ems.polygon USING btree(upper(code));
 82
 83 -- Add geometry field to parent spatial table and spatial index
 84 ALTER TABLE ems.g ADD geom geometry;
 85 CREATE INDEX idx_g_geom ON ems.g USING GIST(geom);
 86
 87 -- Create spatial table to store rejected geometries (other than point, polyline, polygon/multipolygon)
 88 -- Note that this table is not required to be registered in the geometry_columns; as these geometries
 89 -- are rejected, it is assumed that they are must not be accessed through any desktop GIS, which, by the
 90 -- way, do not generally permit loading of multi-geometries data tables
 91 CREATE TABLE ems.gRejects(
 92   id INTEGER PRIMARY KEY,
 93   rDate date NOT NULL,              -- Database recording date
 94   code varchar(150),
 95   geom geometry
 96   );
 97
 98 -- Create rule to redirect parent spatial table features entries to proper child tables or to a reject
 99 -- table for unsupported geometries
100 CREATE OR REPLACE RULE chkins_g AS
101   ON INSERT TO ems.g DO INSTEAD (
102
103     INSERT INTO ems.polygon (id, rDate, code, geom)
104       SELECT new.id, CURRENT_DATE, new.code, st_multi(new.geom) AS geom
105       WHERE geometrytype(new.geom) = 'MULTIPOLYGON' OR geometrytype(new.geom) = 'POLYGON';
106
107     INSERT INTO ems.polyline (id, rDate, code, geom)
108       SELECT new.id, CURRENT_DATE, new.code, new.geom AS geom
109       WHERE geometrytype(new.geom) = 'LINESTRING';
110
111     INSERT INTO ems.point (id, rDate, code, geom)
112       SELECT new.id, CURRENT_DATE, new.code, new.geom AS geom
113       WHERE geometrytype(new.geom) = 'POINT';
114
115     INSERT INTO ems.gRejects(id, rDate, code, geom)
116       SELECT new.id, CURRENT_DATE, new.code, st_multi(new.geom) AS geom
117       WHERE new.geom IS NULL OR geometrytype(new.geom)
```

```
118             NOT IN('POINT', 'LINESTRING', 'POLYGON', 'MULTIPOLYGON');
119   );
120
121 -- Insertion test data for few relevant geometries, including monitoring points, a land use polygon,
122 -- a river, and an unsupported geometry to be rejected to gRejects (temporary) table. Geometries
123 -- from test case in PostGIS in Action.
124 -- Note that, quite differently from previous commented test case (after PostGIS in Action),
125 -- ST_Point could not be used and SRID has been defined as a parameter in all ST_GeomFromText calls.
126 INSERT INTO ems.g(code, geom)
127 VALUES ('Mp01', ST_GeomFromText('POINT(0.1 59)',4326)),
128        ('Mp02', ST_GeomFromText('POINT(0.2 59.2)',4326)),
129        ('Mp03', ST_GeomFromText('POINT(0.3 59.1)',4326)),
130        ('LandUse01', ST_GeomFromText('POLYGON ((0.123 59.034,
131           0.125 59.337, 0.258 59.054, 0.280 58.999, 0.123 59.034))',4326)),
132        ('LandUse02', ST_GeomFromText('POLYGON ((0.123 59.102,
133           0.224 59.205, 0.225 59.205, 0.176 59.105, 0.123 59.102))',4326)),
134        ('River01', ST_GeomFromText('LINESTRING(0.127 58.95, 0.228 59.82)',4326) ),
135        ('CircularString', ST_GeomFromText('CIRCULARSTRING(0 0, 2 0, 2 2, 0 2, 0 0)',4326));
136
137 --
138 -- Creation of time series tables hierarchy
139 --
140
141 -- Parent time series table
142 CREATE TABLE ems.ts(
143   id serial PRIMARY KEY,
144   gId integer NOT NULL,             -- FK referencing PK of geometry
145   tsTypeId integer NOT NULL,        -- FK referencing PK of time series type (tsType table)
146   rDate date NOT NULL,              -- Database recording date
147   mDate date NOT NULL,              -- Measurement date
148   m double precision NOT NULL       -- Measurement
149   );
150
151 -- Child table for time series referring to point features
152 CREATE TABLE ems.tsPoint(
153   CONSTRAINT pk_tsPoint PRIMARY KEY (id))
154   INHERITS (ems.ts);
155
156 -- Child table for time series referring to polyline features
157 CREATE TABLE ems.tsPolyline(
158   CONSTRAINT pk_tsPolyline PRIMARY KEY (id))
159   INHERITS (ems.ts);
```

```
160
161 -- Child table for time series referring to polygon features
162 CREATE TABLE ems.tsPolygon(
163   CONSTRAINT pk_tsPolygon PRIMARY KEY (id))
164   INHERITS (ems.ts);
165
166 -- Create btree indexes on measurement date (mDate) for each child table
167 CREATE INDEX idx_tsPoint_mDate
168 ON ems.tsPoint USING btree(mDate);
169
170 CREATE INDEX idx_tsPolyline_mDate
171 ON ems.tsPolyline USING btree(mDate);
172
173 CREATE INDEX idx_tsPolygon_mDate
174 ON ems.tsPolygon USING btree(mDate);
175
176 -- Create tsRejects table to store rejected data measurements (i.e. uncorrect records
177 -- referencing a non existing geometry)
178 CREATE TABLE ems.tsRejects(
179   id INTEGER PRIMARY KEY,
180   gId integer NOT NULL,             -- FK referencing PK of geometry
181   tsTypeId integer NOT NULL,        -- FK referencing PK of time series type (tsType table)
182   rDate date NOT NULL,              -- Database recording date
183   mDate date NOT NULL,              -- Measurement date
184   m double precision NOT NULL       -- Measurement
185   );
186
187 -- Test queries on geographical data tables hierarchy through top parent table
188 SELECT * FROM ems.g;
189
190 SELECT id, code, geometrytype(geom) FROM ems.g
191   WHERE code = 'Mp02';
192
193 SELECT id, code, geom, geometrytype(geom) FROM ems.g
194   WHERE geometrytype(geom) = 'POINT';
195
196 SELECT id, code, geom, geometrytype(geom) FROM ems.g
197   WHERE geometrytype(geom) IN ('POINT','LINESTRING');
198
199 SELECT id, code, geom, geometrytype(geom) FROM ems.g
200    WHERE geometrytype(geom) IN ('POINT') AND code='Mp01';
201
```

```
202 -- Create rule to redirect parent time series data to proper child tables (or to a
203 -- reject table for uncorrect data, i.e. data which would refer to non existing
204 -- geographical objects)
205 CREATE OR REPLACE RULE chkins_ts AS
206   ON INSERT TO ems.ts DO INSTEAD (
207
208     INSERT INTO ems.tsPolygon (id, gId, tsTypeId, rDate, mDate, m)
209       SELECT new.id, new.gId, new.tsTypeId, CURRENT_DATE, new.mDate, new.m
210       WHERE EXISTS (
211         SELECT * FROM ems.g where id = new.gId AND
212                                 geometrytype(geom) IN ('POLYGON','MULTIPOLYGON'));
213
214     INSERT INTO ems.tsPolyline (id, gId, tsTypeId, rDate, mDate, m)
215       SELECT new.id, new.gId, new.tsTypeId, CURRENT_DATE, new.mDate, new.m
216       WHERE EXISTS (
217         SELECT * FROM ems.g where id = new.gId AND
218                                 geometrytype(geom) = 'LINESTRING');
219
220     INSERT INTO ems.tsPoint (id, gId, tsTypeId, rDate, mDate, m)
221       SELECT new.id, new.gId, new.tsTypeId, CURRENT_DATE, new.mDate, new.m
222       WHERE EXISTS (
223         SELECT * FROM ems.g where id = new.gId AND
224                                 geometrytype(geom) = 'POINT');
225
226     -- No data with given id are available among geometries
227     INSERT INTO ems.tsRejects (id, gId, tsTypeId, rDate, mDate, m)
228       SELECT new.id, new.gId, new.tsTypeId, CURRENT_DATE, new.mDate, new.m
229       WHERE NOT EXISTS (SELECT * FROM ems.g where id = new.gId);
230   );
231
232 -- Add foreign key constraints in child time series tables, pointing at related
233 -- geometry tables
234 ALTER TABLE ems.tsPoint
235 ADD CONSTRAINT fk_tsPoint_Point
236 FOREIGN KEY (gId)
237 REFERENCES ems.Point (id)
238 ON UPDATE CASCADE ON DELETE RESTRICT;
239
240 ALTER TABLE ems.tsPolyline
241 ADD CONSTRAINT fk_tsPolyline_Polyline
242 FOREIGN KEY (gId)
243 REFERENCES ems.Polyline (id)
```

```
244 ON UPDATE CASCADE ON DELETE RESTRICT;
245
246 ALTER TABLE ems.tsPolygon
247 ADD CONSTRAINT fk_tsPolygon_Polygon
248 FOREIGN KEY (gId)
249 REFERENCES ems.Polygon (id)
250 ON UPDATE CASCADE ON DELETE RESTRICT;
251
252 -- Test on time series table insertion
253 -- Note: in current tests, it is assumed that ids of geographical objects (referred
254 -- to through foreign keys) are known; although this would make sense in specific
255 -- cases, this is not usually the case and explicit subquery returning id of an
256 -- existing geographical object must be entered Note that also tsTypeId is not yet
257 -- referencing  a matching record in tsType table, which, by the way, has not been
258 -- created for the moment (see later).
259 -- Take into account that, if other INSERT/DELETE tests have been performed other
260 -- than previously reported, SQL statements here below could not complete as expected;
261 -- when FKs (gIds in following INSERTs) do not refer to existing geometries, records
262 -- are automatically redirected to the tsRejects table. As no error is issued at SQL
263 -- run time, this can potentially be somehow confusing.
264 -- An application should give evidence of such a redirection.
265 -- Check geographical object IDs, before proceeding, based on following SELECT statement
266 SELECT * from ems.g;
267
268 -- Insert data to time series table referencing existing geometries (check as above)
269 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
270   VALUES(15,100,'2010/06/1',12.5);
271 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
272   VALUES(15,100,'2010/06/2',15.5);
273 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
274   VALUES(15,100,'2010/06/3',16.5);
275
276 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
277   VALUES(17,100,'2009/06/1',312.5);
278 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
279   VALUES(17,100,'2009/06/2',315.5);
280
281 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
282   VALUES(18,100,'2009/06/1',1000);
283 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
284   VALUES(18,100,'2009/06/2',2000);
285
```

```
286 -- Insert data to time series table, but referencing a non existing geometry
287 -- (record to be rejected)
288 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
289   VALUES(140,100,'2009/06/1',2.0);
290
291 -- Example of populating time series table, based on a geometry id
292 -- selected after a code (it's assumed code exists and is unique at this stage)
293 -- Note that no unicity constraint has been defined on code attribute, assuming
294 -- that in complex (i.e. multi-company) contexts, the same 'code' could (unfortunately)
295 -- refer to different geometries. More complex constraints should be defined, as that
296 -- no two points could share the same code except if they would be far away each other
297 -- for more than x linear units. Such a distance is a complex function of application
298 -- domain and levels of generalization (a national wide project would not share the same
299 -- snapping distance of a local scale project focused on a contaminated environmental site)
300 -- Note that if geometry does not exist, time series is saved to tsRejects
301 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
302   VALUES(
303     (SELECT id FROM ems.g WHERE code = 'Mp03'),
304     100,
305     '2010/8/1',
306     5000
307   );
308
309 -- Example of populating time series table, based on a geometry id
310 -- selected after a code, which does not exist at the moment.
311 -- This insertion statement results in an error, as a NULL gid violates not-null
312 -- constraint.
313 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
314   VALUES(
315     (SELECT id FROM ems.g WHERE code = 'Dummy'),   -- A non existing code (and geometry)
316     100,
317     '2010/8/1',
318     5000
319   );
320
321 -- Remove data from geographic tables. A more realistic data set is imported at
322 -- a later stage. Note that deletion attempt here below does not work, as
323 -- geographic data have related records in time series ts table
324 DELETE FROM ems.g;
325
326 -- Remove all records from ts (before) and then from g. Note that deletion now works,
327 -- also through child tables
```

```
328 DELETE FROM ems.ts;
329 DELETE FROM ems.g;
330
331 -- Remove rejected records in both geographic and time series tables
332 DELETE from ems.gRejects;
333 DELETE FROM ems.tsRejects;
334
335 -- Check that no more records are available
336 SELECT * FROM ems.g;
337 SELECT id, rDate, code, ST_AsEwkt(geom) FROM ems.g;
338
339 -- Further tests documenting spatial SQL in action as geometry creation,
340 -- SRID (Spatial Reference ID) setting, geometry textual (and extended
341 -- textual) description, and coordinates transformation
342 SELECT ST_SetSRID(ST_Point(343600,2349720),32640);
343 SELECT ST_AsText(ST_SetSRID(ST_Point(343600,2349720),32640));
344 SELECT ST_AsEwkt(ST_SetSRID(ST_Point(343600,2349720),32640));
345 SELECT ST_Transform(ST_SetSRID(ST_Point(343600,2349720),32640),4326);
346 SELECT ST_AsEwkt(ST_Transform(ST_SetSRID(ST_Point(343600,2349720),32640),4326));
347
348 -- Check constraints, based on a point geometry and related time series data
349 -- Insert a new point geometry
350 INSERT INTO ems.g(code, geom)
351   VALUES ('Mp2000',
352           ST_Transform(ST_SetSRID(ST_Point(343600,2349720),32640),4326));
353
354 -- Insert data to time series table referencing point above
355 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
356   VALUES((SELECT id FROM ems.g WHERE code='Mp2000'),100,'2010/06/1',1000);
357 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
358   VALUES((SELECT id FROM ems.g WHERE code='Mp2000'),100,'2010/06/2',1001);
359
360 -- Attempt to delete a geometry with referenced data. This operation fails,
361 -- due to existing constraints
362 DELETE FROM ems.Point where code='Mp2000';
363
364 -- Perform previous operation, provided that data related to geometry are
365 -- removed in advance from tsPoint table (or from its parent table ts)
366 DELETE FROM ems.tsPoint where
367   gid=(SELECT id FROM ems.g WHERE code='Mp2000');
368
369 DELETE FROM ems.Point where code='Mp2000';
```

```
370
371 --
372 -- Creation of time series type table
373 --
374
375 -- Domain to keep trace of time series data types: 1. Instantaneous data 2. Cumulative data
376 -- 3. Incremental data 4.Average data 5.Maximum data 6. Minimum data
377 -- Classification based on ESRI Hydro Data Model
378 CREATE DOMAIN ems.dmDataType INT NOT NULL
379 DEFAULT 1
380 CHECK (VALUE =1 OR     -- Instantaneous data
381        VALUE =2 OR     -- Cumulative data
382        VALUE =3 OR     -- Incremental data
383        VALUE =4 OR     -- Average data
384        VALUE =5 OR     -- Maximum data
385        VALUE =6);      -- Minimum data
386
387 -- Domain to keep trace of time series origin: 1. Observed 2. Computed (i.e. from a model)
388 -- Arc Hydro data model uses the analogous concepts of Recorded/Generated
389 CREATE DOMAIN ems.dmOrigin INT NOT NULL
390 DEFAULT 1
391 CHECK (VALUE = 1 OR    -- Observed
392        VALUE = 2);     -- Computed
393
394 -- Creation of TsType table
395 CREATE TABLE ems.tsType(
396   id serial PRIMARY KEY,
397   code varchar(50) NOT NULL,      -- tsType code identifying time series (i.e. 'T' for a generic Temperature,
398                                   -- to refer to all measured data, 'T-Sim01' to refer to T in Sim01)
399   variable varchar(50) NOT NULL,  -- Variable description (i.e. Temperature)
400   units varchar(50) NOT NULL,     -- Measurement units (i.e. mm/yr)
401   dataType ems.dmDataType,        -- Data type (DEFAULT 1 for istantaneous)
402   origin ems.dmOrigin             -- Data origin (DEFAULT 1 for Observed)
403   );
404
405 -- Add constraints on tsType table
406 ALTER TABLE ems.tsType ADD CONSTRAINT code_unique UNIQUE(code);
407 ALTER TABLE ems.tsType ADD CONSTRAINT symbol_unique UNIQUE(code,variable,units,dataType,origin);
408
409 -- Add foreign key constraints in child time series tables
410 ALTER TABLE ems.tsPoint
411 ADD CONSTRAINT fk_tsPoint_tsType
```

```
412 FOREIGN KEY (tsTypeId)
413 REFERENCES ems.tsType(id)
414 ON UPDATE CASCADE ON DELETE RESTRICT;
415
416 ALTER TABLE ems.tsPolyline
417 ADD CONSTRAINT fk_tsPolyline_tsType
418 FOREIGN KEY (tsTypeId)
419 REFERENCES ems.tsType(id)
420 ON UPDATE CASCADE ON DELETE RESTRICT;
421
422 ALTER TABLE ems.tsPolygon
423 ADD CONSTRAINT fk_tsPolygon_tsType
424 FOREIGN KEY (tsTypeId)
425 REFERENCES ems.tsType(id)
426 ON UPDATE CASCADE ON DELETE RESTRICT;
427
428 -- Original ESRI Hydro Data model isRegular and tsInterval attributes, to keep trace of
429 -- regularly recorded data, have been voluntary kept out of previous model of tsType;
430 -- they would add additional complexity, while this kind of information can be derived
431 -- from data set.
432
433 -- Input test records involving all tables and check for errot when attempting to
434 -- populate time series ts table
435 INSERT INTO ems.tsType (code,variable,units)
436   VALUES('T','Temperature','°C');
437
438 INSERT INTO ems.tsType (code,variable,units,origin)
439   VALUES('T-Sim01','Temperature','°C',2);
440
441 INSERT INTO ems.g(code, geom)
442   VALUES ('Mp2000',
443           ST_Transform(ST_SetSRID(ST_Point(343600,2349720),32640),4326));
444
445 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
446   VALUES((SELECT id FROM ems.g WHERE code='Mp2000'),
447          (SELECT id FROM ems.tsType WHERE code='T'),
448          '2010/06/1',
449          10);
450
451 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
452   VALUES((SELECT id FROM ems.g WHERE code='Mp2000'),
453          (SELECT id FROM ems.tsType WHERE code='T'),
```

```
454          '2010/06/2',
455          20);
456
457 -- Demonstrate error in populating ts table, due to lack of referenced id in tsType
458 INSERT INTO ems.ts(gId,tsTypeId,mDate,m)
459   VALUES((SELECT id FROM ems.g WHERE code='Mp2000'),
460          2500,
461          '2010/06/2',
462          20);
463
464 -- Creation of views
465 -- Note: "ts.id as id" is required. When accessing data from a deskop GIS as QuantumGIS,
466 -- a PK named 'id' is looked for
467 CREATE OR REPLACE VIEW ems.pointView AS (
468   SELECT ts.id as id,
469          p.code,
470          -- geometrytype(p.geom), -- It is not required
471          p.geom,
472          p.rDate,
473          ts.mDate,
474          ts.m
475   FROM ems.point p, ems.tsPoint ts
476   WHERE p.id = ts.gid);
477
478 CREATE OR REPLACE VIEW ems.polylineView AS (
479   SELECT ts.id as id,
480          p.code,
481          -- geometrytype(p.geom), -- It is not required
482          p.geom,
483          p.rDate,
484          ts.mDate,
485          ts.m
486   FROM ems.polyline p, ems.tsPolyline ts
487   WHERE p.id = ts.gid);
488
489 CREATE OR REPLACE VIEW ems.polygonView AS (
490   SELECT ts.id as id,
491          p.code,
492          -- geometrytype(p.geom), -- It is not required
493          p.geom,
494          p.rDate,
495          ts.mDate,
```

```
496          ts.m
497   FROM ems.polygon p, ems.tsPolygon ts
498   WHERE p.id = ts.gid);
499
500 -- Views above must be registered manually in geometry_columns metadata
501 -- See 4.3.4. Manually Registering Geometry Columns in geometry_columns
502 -- http://postgis.refractions.net/docs/ch04.html#Manual_Register_Spatial_Column
503 -- Register this table in AddGeometry columns - do the following
504 INSERT INTO geometry_columns
505 (f_table_catalog, f_table_schema, f_table_name, f_geometry_column, coord_dimension, srid, "type")
506 SELECT '', 'ems', 'pointview', 'geom', ST_CoordDim(geom), ST_SRID(geom), GeometryType(geom)
507 FROM ems.pointview LIMIT 1;
508
509 INSERT INTO geometry_columns
510 (f_table_catalog, f_table_schema, f_table_name, f_geometry_column, coord_dimension, srid, "type")
511 SELECT '', 'ems', 'polylineView', 'geom', ST_CoordDim(geom), ST_SRID(geom), GeometryType(geom)
512 FROM ems.pointview LIMIT 1;
513
514 INSERT INTO geometry_columns
515 (f_table_catalog, f_table_schema, f_table_name, f_geometry_column, coord_dimension, srid, "type")
516 SELECT '', 'ems', 'polygonView', 'geom', ST_CoordDim(geom), ST_SRID(geom), GeometryType(geom)
517 FROM ems.pointview LIMIT 1;
518
519 -- ********************************************************************************************
520 -- Case study
521 -- Loading data from csv files to temporary tables. Csv files have been exported after shape files,
522 -- with projected coordinates in WGS84 UTM 40N, located in south Iran.
523 -- Recall that a shape file loader is available within PgAdmin III
524 -- Important note: PK constraint on Point table does not pose any problem when populating table from
525 -- text file (see later) but a potential problem arises when you rely on automatic PK generation,
526 -- which is defined as serial. In this case already existing value could be generated, leading to
527 -- a duplication key conflict, to be properly managed (i.e. by developing a specific trigger)
528 CREATE TABLE ems.tmpWell (
529   id integer PRIMARY KEY,
530   code varchar(20),
531   X float,
532   y float
533 );
534
535 -- Copy data from a comma delimited temporary Well table. Note that reference points to a
536 -- location on server (not on client machine)
537 /*
```

```
538 COPY ems.tmpWell FROM
539
'C://Ezio//Formazione//Hertfordshire//MScComputerScience//Courses//MScDissertation//Application//Data//tmpWell.txt'
540   DELIMITER ',';
541 */
542
543 -- Copy data from the comma delimited temporary Well table. Note that this command must be run
544 -- from plsql, referring to a path on client machine (contrasting to above SQL statement)
545 \copy ems.tmpWell from
546
'C://Ezio//Formazione//Hertfordshire//MScComputerScience//Courses//MScDissertation//Application//Data//CaseStudy//txt//
well.txt'
547 DELIMITER AS ','
548
549 -- Note this is not strictly required, as the table is only temporary
550 vacuum analyze ems.tmpWell;
551
552 -- Remove data from point table, if any
553 DELETE FROM ems.point;
554
555 -- Query spatial reference table to access srid of WGS84 UTM40N of case study example
556 SELECT srid, srtext,proj4text
557 FROM spatial_ref_sys
558 WHERE proj4text ILIKE '%nad83%'
559 AND proj4text ILIKE '%grs80%' AND proj4text ILIKE '%utm%';
560
561 -- WGS84 UTM40N SRID is 32640
562 SELECT srid, srtext,proj4text
563 FROM spatial_ref_sys
564 WHERE proj4text ILIKE '%UTM%' AND proj4text ILIKE '%zone=40%';
565
566 -- Remove all geographic data and related time series data (these last ones in advance,
567 -- otherwise a constraint violation is issued
568 DELETE FROM ems.ts;
569 DELETE FROM ems.g;
570
571 -- Transfer data from the temporary monitoring well table to the spatial g table
572 -- NOTE: PK constraint should be temporarily removed, as PK is currently
573 -- automatically generated after a sequence (serial). However, despite the original
574 -- PK definition as a serial field, this choice can be overriden within an INSERT
575 -- statement by explicitly populating the id. Further record insertions should be
576 -- properly managed in order to avoid that already existing ids could generate
```

```
577 -- duplication conflicts.
578 -- This is a general relevant issue, when both IDs automatic generation and explicit
579 -- definition are used
580 INSERT INTO ems.g(id,code,geom)
581   SELECT id,code, ST_Transform(ST_SetSRID(ST_MakePoint(x,y),32640),4326) As geom
582   FROM ems.tmpWell;
583
584 -- Select point geographic data including an extended (with SRID) WKT description
585 SELECT id, rdate, code, ST_AsEwkt(geom)
586 FROM ems.point
587 ORDER BY id;
588
589 -- Populate time series type table
590 -- Note that, as PK is automatically generated, time series data imported from
591 -- external tables must be consistent as far as is concerned their foreign key
592 -- referencing measurement type.
593 -- In this specific case study, where tsType are assumed to start from 1, following
594 -- actions could be put in place:
595 --   1. tsType table must be removed and recreated (this implies removing/recreating
596 --      constraints)
597 --   2. tsType table must be populated as follows
598 -- Alternatively serial generation of PK should be removed and the field id explicitly
599 -- populated at this stage. Apparently dropping the automatic generation of PK is not
600 -- feasible. A final point is that variable names and measurement units must be set
601 -- consistently with the the original data set.
602 -- See http://neilconway.org/docs/sequences/ for the PostgreSQL sequences
603 -- http://developer.mimer.com/documentation/html_93/Mimer_SQL_Mobile_DocSet/SQL_Statements9.html
604 DELETE FROM ems.tsType;
605
606 -- Tests to access current value in generating sequence and to alter it
607 -- Access current value in sequence. Note this SQL statement rises the error "curval of sequence .. is
608 -- not yet defined in this session". This is apparently due to the fact that this statement must be
609 -- run only when SELECT CURRVAL statement has been executed. See http://code.djangoproject.com/ticket/9302
610 SELECT currval(pg_get_serial_sequence('ems.tsType', 'id'));
611
612 -- Alter starting value in sequence (name is currently generated by PostGIS as the PK id is defined
613 -- as serial). Note that the sequence must not be in use, when trying to alter it
614 ALTER SEQUENCE ems.tstype_id_seq RESTART WITH 1;
615
616 -- Populate tsType table with variables symbol, extended definition and measurement units
617 INSERT INTO ems.tsType (code,variable,units)
618   VALUES
```

```
    ('h','Piezometric head','m a.s.l.'),
    ('v2','v2','n.a.'),
    ('v3','v3','n.a.'),
    ('v4','v4','n.a.'),
    ('v5','v5','n.a.'),
    ('v6','v6','n.a.'),
    ('v7','v7','n.a.'),
    ('v8','v8','n.a.'),
    ('v9','v9','n.a.'),
    ('v10','v10','n.a.'),
    ('v11','v11','n.a.'),
    ('v12','v12','n.a.'),
    ('v13','v13','n.a.'),
    ('v14','v14','n.a.'),
    ('v15','v15','n.a.'),
    ('v16','v16','n.a.'),
    ('v17','v17','n.a.'),
    ('v18','v18','n.a.'),
    ('v19','v19','n.a.'),
    ('v20','v20','n.a.'),
    ('v21','v21','n.a.'),
    ('v22','v22','n.a.'),
    ('v23','v23','n.a.'),
    ('v24','v24','n.a.'),
    ('v25','v25','n.a.'),
    ('v26','v26','n.a.'),
    ('v27','v27','n.a.');

-- Clean tsRejects, keeping trace of records conflicting with current constraints
DELETE FROM ems.tsRejects;

-- Populating time series ts table
--
-- Import time series data to ems.ts table. Given all above considerations, import has been performed
-- by creating programmatically an INSERT SQL statement, in order to fully exploit automatic generation
-- of table PK and of current (recording) date. Due to intrinsic limitations of both SQL COPY and plsql
-- \copy statements, they have been both discarded as viable options.
--
-- The VBA code to create INSERT SQL statements is reported in appendix
-- The code generates a file named insert_tmpts.sql, which must be loaded and run in SQL editor in
-- order to load data to ts table
---------------- FINISHED --------------------
```

```
661
662
663
664 -- ********************************************************************************************
665 -- Additional code to remove constraints and tables, following best practice guidelines. This code is
666 -- tipically used to address the requirement to rebuild the schema from scratch.
667 -- Drop all tables if already existing. This is required in case the system architecture must be revised.
668 -- Particularly geometric tables must be dropped by using the DropGeometryTable command in order to
669 -- guarantee that related entry in geometry_columns table is removed
670 -- ********************************************************************************************
671 DROP RULE chkins_g ON ems.g CASCADE;
672
673 ALTER TABLE ems.tspoint DROP CONSTRAINT fk_tspoint_point;
674 SELECT DropGeometryTable('ems','point');
675
676 ALTER TABLE ems.tspolyline DROP CONSTRAINT fk_tspolyline_polyline;
677 SELECT DropGeometryTable('ems','polyline');
678
679 ALTER TABLE ems.tspolygon DROP CONSTRAINT fk_tspolygon_polygon;
680 SELECT DropGeometryTable('ems','polygon');
681
682 DROP RULE chkins_ts ON ems.ts CASCADE;
683 DROP TABLE ems.tspoint;
684 DROP TABLE ems.tspolyline;
685 DROP TABLE ems.tspolygon;
686
687 DROP TABLE ems.g;
688 DROP TABLE ems.grejects;
689 DROP TABLE ems.ts;
690 DROP TABLE ems.tsrejects;
691 DROP TABLE ems.tstype;
```

APPENDIX 3 – GOOGLE MAPS MASHUP: HTML, CSS AND JAVA SCRIPT SOURCE CODE

index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Environmental Monitoring System</title>
    <link type="text/css" href="css/style.css" rel="stylesheet" media="all" />
    <script type="text/javascript" src="http://www.google.com/jsapi"></script>
    <script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=false&amp;language=en"></script>
    <!-- Libraries for application, XmlHttp request and markers clustering -->
    <script type="text/javascript" src="js/map.js"></script>
    <script type="text/javascript" src="js/util.js"></script>
    <script type="text/javascript" src="js/fluster2/lib/Fluster2.packed.js"></script>

  </head>
  <body>
    <!--<h1>Environmental Monitoring System</h1> -->
    <p>
      <select id="list_tstype" onchange="onSelect_list_tstype(this.value)"></select>
      <input type="button" value="Time Series" id="timeSeries" style="width=200px"/>
      <input type="button" value="Clear" id="clear"/>
    </p>
    <div id="map"></div>
    <!-- div id="chart" style="width:100%; height:300px"></div>  -->
    <div id="chart" style="height:50%"></div>
  </body>
</html>
```

style.css

```
/*
Author: Ezio Crestaz
Email: ezio.crestaz@giscience.it
```

```
 4 Created: 2011-01-13
 5 Description: stylesheet for environmental monitoring mashup
 6 */
 7 html, body {
 8   height: 92%;
 9   font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
10   font-size: small;
11   background: #fff;
12 }
13 #map {
14   width: 100%;
15   height: 92%;    // Note that also html and body height must be set
16   border: 1px solid #000;
17   position: absolute;
18 }
19 #chart {
20   width: 50%;
21   position: relative;   // Position according to parent div
22   top: 0px;
23   left: 50%;
24
25  /*Use this for your trasparency*/
26   /*
27   background-color: none;
28   background-image: none;
29   opacity: .5; */   /*for real browsers (Chrome, Firefox, Safari..)*/
30   /* filter:alpha(opacity=60); */ /* for IE */
31
32 }
```

map.js

```
 1 /*
 2  * Author: Ezio Crestaz
 3  * Date: January 30th, 2011
 4  * Scope: function controlling Google Maps v. 3 mapping application, managing environmental
 5  *        monitoring data and related time series stored to remote PostgreSQL/PostGIS database.
 6  *        Time series are visualized within an 'annotatedtimeline' Google widget
 7 */
 8
 9 // Anonymous function which is also immediately run to build and manage all web
```

```
// mapping application objects
(function() {
  // Load Google annotated time line visualisation package
  google.load("visualization", "1", {packages: ["annotatedtimeline"]});

  window.onload = function() {
    // Load Google annotatedtimeline library to build time-dependent graphs
    //google.load("visualization", "1", {packages: ["annotatedtimeline"]});

    // Extend Array object with contains and remove functions.
    // Contains function returns a boolean, depending upon obj existence within the array
    // Remove function deletes obj element (if existent) from array
    // http://stackoverflow.com/questions/237104/javascript-array-containsobj
    Array.prototype.contains = function(obj) {
      var i = this.length;
      while (i--) {
        if (this[i] === obj) {
          return true;
        }
      }
      return false;
    }

    Array.prototype.remove = function(obj){
      var i = this.length;
      while (i--) {
        if(obj==this[i]) this.splice(i, 1);
      }
    }

    // Variables containing references to html elements to visualise map, chart, coordinates
    // of initial reference point, map bounding rectangle, markers array, selected points array,
    // infowindow and fluster for point clustering

    var mapDiv = document.getElementById('map');
    var chartDiv  = document.getElementById('chart');
    var latlng = new google.maps.LatLng(0.0, 0.0);
    var bounds = new google.maps.LatLngBounds();
    var selected = new Array();          // Selected markers codes
    var selectedMarkers = new Array();   // Selected markers (only to reset their icons on clear)
    var infowindow;
    var fluster;
```

```
52
53      // Symbol for groundwater observation point after UNESCO hydrogeological legend (1970)
54      // redrawn by http://commons.wikimedia.org/wiki/Category:Symbols_for_hydrogeological_maps
55      // Credits: sample custom marker code created with Google Map Custom Marker Maker
56      // http://www.powerhut.co.uk/googlemaps/custom_markers.php
57      // Image variable constructor requires a reference to a graphical file, image size,
58      // origin within sprite (currently not used) and anchor point
59      // Note: single images should be substituted with a sprite at a later stage, to improve
60      // application performance
61
62      var image = new google.maps.MarkerImage(
63        'marker-images/gwobs.png',
64        new google.maps.Size(32,32),
65        null,                           //new google.maps.Point(0,0),
66        new google.maps.Point(16,16)
67      );
68
69      var shadow = new google.maps.MarkerImage(
70        'marker-images/gwobs_shadow.png',
71        new google.maps.Size(52,32),
72        null,                           //new google.maps.Point(0,0),
73        new google.maps.Point(16,16)
74      );
75
76      // Image for selected points
77      var image_selected = new google.maps.MarkerImage(
78        'marker-images/gwobs_selected.png',    //gwobs.png',
79        new google.maps.Size(32,32),
80        null,                            //new google.maps.Point(0,0)
81        new google.maps.Point(16,16)
82      );
83
84      // Shape defines clickable area
85      // Note: polygon could be substitued with a circle
86      var shape = {
87        type: 'poly',
88        coord: [19,0,21,1,23,2,25,3,26,4,27,5,28,6,28,7,29,8,29,9,30,10,
89                30,11,30,12,31,13,31,14,31,15,31,16,31,17,31,18,30,19,30,
90                           20,30,21,29,22,29,23,28,24,28,25,27,26,26,27,25,28,23,29,
91                           21,30,18,31,13,31,10,30,8,29,6,28,5,27,4,26,3,25,3,24,2,23,2,
92                           22,1,21,1,20,1,19,0,18,0,17,0,16,0,15,0,14,0,13,1,12,1,11,1,
93                           10,2,9,2,8,3,7,3,6,4,5,5,4,6,3,8,2,10,1,12,0,19,0]
```

```
94      };
95
96      // Create a map options literal
97      var options = {
98        center: latlng,
99        zoom: 4,
100       scaleControl: true,
101       mapTypeId: google.maps.MapTypeId.TERRAIN,    // ROADMAP
102       draggableCursor: 'move',
103       draggingCursor: 'move',
104       mapTypeControl: true,
105       mapTypeControlOptions: {
106         position: google.maps.ControlPosition.LEFT,
107         style: google.maps.MapTypeControlStyle.DROPDOWN_MENU
108       }
109     };
110
111     // Create the map
112     var map = new google.maps.Map(mapDiv, options);
113     // Create the chart
114     var chart = new google.visualization.AnnotatedTimeLine(chartDiv);
115
116     // Populate time series list
117     // Code from http://www.electrictoolbox.com/javascript-add-options-html-select/
118     var $url = "http://159.149.84.152/crestaz/ems/appl/php/tstype.php";
119     var select=document.getElementById('list_tstype');
120     downloadUrl($url, function(data2) {
121             var tstypes = data2.documentElement.getElementsByTagName("tstype");
122             for (var i = 0; i < tstypes.length; i++) {
123         var id = tstypes[i].getAttribute("id");
124         var code = tstypes[i].getAttribute("code");
125         select.options[select.options.length] = new Option(code, id);
126             }
127           });
128
129     // Download XML data locations from server, based on a query managed by the dedicated PHP program
130           downloadUrl("http://159.149.84.152/crestaz/ems/appl/php/gmpg.php", function(data) {
131
132       var markers = data.documentElement.getElementsByTagName("marker");
133
134       for (var i = 0; i < markers.length; i++) {
135         var latlng = new google.maps.LatLng(parseFloat(markers[i].getAttribute("lat")),
```

```
                                        parseFloat(markers[i].getAttribute("lon")));
        var code = markers[i].getAttribute('code');
        var marker = new google.maps.Marker({
                position: latlng,
                map: map,
                title: code,
          draggable: false,
          icon: image,
          shadow: shadow,
          shape: shape,
          flat: true
              });

        // Wrap mouse clidk event listeners inside an anonymous function, which is immediately call
        (function(marker,code) {
           google.maps.event.addListener(marker, 'click', function() {
             // Create a info window with a map which is maximized
             // Code after Svennerberg, 2010, p. 149-150
             var detailDiv = document.createElement('div');
             detailDiv.style.width = '200px';
             detailDiv.style.height = '200px';
             document.getElementById('map').appendChild(detailDiv);

             // Create MapOptions for the overview map
             var overviewOpts = {
               zoom: 14,
               center: marker.getPosition(),
               mapTypeId: map.getMapTypeId(),
               disableDefaultUI: true
             };

             var detailMap = new google.maps.Map(detailDiv, overviewOpts);

             // Create a marker that will show in the detail map
             var detailMarker = new google.maps.Marker({
               position: marker.getPosition(),
               map: detailMap,
               clickable: false
             });

             if (!infowindow) {
               infowindow = new google.maps.InfoWindow();
```

```
178             }
179             //infowindow.setContent(code);  // Code is also visualized in Tooltip
180             infowindow.setContent(detailDiv);
181             infowindow.open(map, marker);
182 
183                 // Set marker icon to selected and updates selected pmonitoring points array
184                   if (!selected.contains(code)) {
185                     marker.setIcon(image_selected);
186                     selected.push(code);
187                     selectedMarkers.push(marker);
188                   } else {
189                     marker.setIcon(image);
190                     selected.remove(code);
191                     selectedMarkers.remove(marker);
192             }
193           })
194 
195           // Set mouse over event, raising up marker icon and revealing shadow
196           google.maps.event.addListener(marker, 'mouseover', function() {
197             this.setFlat(false);
198           })
199 
200           // Set mouse out event, placing marker back on the map
201           google.maps.event.addListener(marker, 'mouseout', function() {
202             this.setFlat(true);
203           })
204 
205         })(marker,code);
206 
207         // Add marker to the Fluster
208               fluster.addMarker(marker);
209                     bounds.extend(latlng);
210       }  // Close 'markers' for loop
211 
212             // Set the map to fit markers bounds
213       map.fitBounds(bounds);
214 
215       // Set default styles for different clusters
216             fluster.styles = {
217               // More than 0 markers
218               0: {
```

```
219                     image: 'http://gmaps-utility-
library.googlecode.com/svn/trunk/markerclusterer/1.0/images/m1.png',
220                       textColor: '#FFFFFF',
221                       width: 53,
222                       height: 52
223                   },
224                 // More than 10 markers
225                 10: {
226                       image: 'http://gmaps-utility-
library.googlecode.com/svn/trunk/markerclusterer/1.0/images/m2.png',
227                       textColor: '#FFFFFF',
228                       width: 56,
229                       height: 55
230                 },
231                 // More than 20 markers
232                 20: {
233                       image: 'http://gmaps-utility-
library.googlecode.com/svn/trunk/markerclusterer/1.0/images/m3.png',
234                       textColor: '#FFFFFF',
235                       width: 66,
236                       height: 65
237                 }
238               };
239
240               // Initialize Fluster, which sets event handlers on the map and calculates clusters the first time.
241               fluster.initialize();
242
243         // Attach click events to the buttons
244         document.getElementById('timeSeries').onclick = function() {
245
246                       // Load and populate time diagram
247           // Note: function operates only once all axyncronous functions below terminated
248                       function loadChart(currentXML) {
249                               if (currentXML == selected.length) {
250               // Join all data tables in the array to build a unique final data table
251                     var dtFinal = dt[0];
252                     var cols =[];   // Array to store columns number to migrate to final data table
253                     for (i=1; i<dt.length; i++) {
254                       for (j=1; j<=i; j++) cols.push(j);
255                       dtFinal = google.visualization.data.join(dtFinal, dt[i], 'full', [[0,0]], cols, [1]);
256                             // http://www.joeyjavas.com/2007/08/10/how-to-remove-all-elements-from-a-javascript-
array/
```

```
257                     cols.length = 0;    // Remove all elements from cols array
258                   }
259                 chart.draw(dtFinal, {displayAnnotations: true, scaleType: 'maximized'});
260           }
261                     }
262
263         if (!selected.length) {
264           alert('No monitoring point selected! Mouse click on points to select');
265         } else {
266           var dt = [];       // Array to store all data tables
267           var nXML = 0;      // Number of XML files fully downloaded
268           var tsTypeList=document.getElementById('list_tstype')  // tsType list
269
270           // Extract information about currently selected time series type
271           for (var k=0; k < tsTypeList.options.length; k++){
272             if (tsTypeList.options[k].selected==true){
273               var tsText = tsTypeList.options[k].text;
274               var tsValue = tsTypeList.options[k].value;
275               break;
276             }
277           }
278
279           // Loop through all selected monitoring points
280           for (i=0; i < selected.length; i++){
281             // Build url remote request for specif point code and time series type
282             $url = 'http://159.149.84.152/crestaz/ems/appl/php/timeseries.php';
283                   $url = $url + "?code="+selected[i];
284                   $url = $url + "&tstypeid=";
285                   $url = $url + tsValue;
286
287             // Anonymous function encapsulating call to XML file downloading to avoid problems related to closures
288                               (function(current_i){
289
290                                       downloadUrl($url, function(data1) {
291
292                                               var dtTemp = new google.visualization.DataTable();    // Variable to
hold a data table;
293                                               dtTemp.addColumn('date', 'Date');
294                                               dtTemp.addColumn('number', tsText + '_' + selected[current_i]);   //
Composed column name (variable+code)
295
296                                               var measures = data1.documentElement.getElementsByTagName("measure");
```

```
297                                             // Extract each measure from time series XML file and populates data
chart
298                                             for (var j = 0; j < measures.length; j++) {
299                                                     var mdate = measures[j].getAttribute("mdate");
300                                                     var value = measures[j].getAttribute("value");
301
302                                                     mdate = mdate.replace(/\D/g, " ");
303                                                     var dObj = mdate.split(" ");
304
305                                                     // Month numbering ranges through 0..11 in jscript, so dates
retrieved from
306                                                     // spatial database are corrected consistently. Modification
of original PostGIS
307                                                     // database could address dates with timestamps, jscript
variable being set as following
308                                                     // var myDate = new Date(dObj[0], (dObj[1]-1), dObj[2],
dObj[3], dObj[4], dObj[5]);
309                                                     var myDate = new Date(dObj[0], (dObj[1]-1), dObj[2]);
310                                                     //document.write("Making a date from a string: " + myDate);
311
312                                                     dtTemp.addRow([myDate, parseFloat(value)]);
313                                             }    // 'Measures' loop end
314
315                                             // Store current data table to array
316                                             dt.push(dtTemp);
317                                             nXML++;   // Increment number of XML files already downloaded
318
319                                             loadChart(nXML);
320                                     })    // Download single time series XML file end
321                             })(i); // End of encapsulating anonynous function
322
323           }  // End loop on selected points
324
325         }
326
327       }
328
329       document.getElementById('clear').onclick = function() {
330         if (!selected.length) {
331           alert('No monitoring point selected! So no monitoring point to clear');
332         } else {
333           // Reset both arrays of selected codes and markers, further to marker symbol
```

```
            selected.length = 0;    // Remove all elements from array of selected points
            for (i=0; i < selectedMarkers.length; i++) {
              selectedMarkers[i].setIcon(image);
            }
            selectedMarkers.length = 0;

          }
        }


      });  // Close downloadUrl function call


      // Initialize Fluster to cluster point data
            fluster = new Fluster2(map);

          }     // Close function attached to window onload event
})();  // Call outer function
```

util.js

```
/**
  * Author: Project hosted on Google code at following web site:
  * http://code.google.com/p/wtracks/source/browse/trunk/common/js/util.js?spec=svn30&r=30
  * Date: downloaded in December 2010
  * Scope: returns an XMLHttp instance to use for asynchronous
  *        downloading. This method will never throw an exception, but will
  *        return NULL if the browser does not support XmlHttp for any reason.
  * @return {XMLHttpRequest|Null}
  */
function createXmlHttpRequest() {
  try {
    if (typeof ActiveXObject != 'undefined') {
          return new ActiveXObject('Microsoft.XMLHTTP');
        } else if (window["XMLHttpRequest"]) {
          return new XMLHttpRequest();
        }
  } catch (e) {
    changeStatus(e);
```

```
19    }
20    return null;
21  };
22
23  /**  * This functions wraps XMLHttpRequest open/send function.
24    * It lets you specify a URL and will call the callback if
25    * it gets a status code of 200.
26    * @param {String} url The URL to retrieve
27    * @param {Function} callback The function to call once retrieved.
28    */
29  function downloadUrl(url, callback) {
30    var status = -1;
31    var request = createXmlHttpRequest();
32    if (!request) {
33      return false;
34    }
35
36    request.onreadystatechange = function() {
37      if (request.readyState == 4) {
38            try {
39              status = request.status;
40            } catch (e) {
41              // Usually indicates request timed out in FF.
42            }
43            if (status == 200) {
44              callback(request.responseXML, request.status);
45                  request.onreadystatechange = function() {};
46            }
47          }
48    }
49    request.open('GET', url, true);
50    try {
51      request.send(null);
52    } catch (e) {
53      changeStatus(e);
54    }
55  };
56
57  /**
58    * Parses the given XML string and returns the parsed document in a
59    * DOM data structure. This function will return an empty DOM node if
```

```
60    * XML parsing is not supported in this browser.    * @param {string} str XML string.
61    * @return {Element|Document} DOM.
62    */
63  function xmlParse(str) {
64    if (typeof ActiveXObject != 'undefined' && typeof GetObject != 'undefined') {
65      var doc = new ActiveXObject('Microsoft.XMLDOM');
66          doc.loadXML(str);
67          return doc;
68    }
69    if (typeof DOMParser != 'undefined') {
70      return (new DOMParser()).parseFromString(str, 'text/xml');
71    }
72    return createElement('div', null);
73  }
74
75  /**    `
76    * Appends a JavaScript file to the page.
77    * @param {string} url
78    */
79  function downloadScript(url) {
80    var script = document.createElement('script');
81    script.src = url;
82    document.body.appendChild(script);
83  }
```

APPENDIX 4 – PHP SERVER SIDE SOURCE CODE

db_credentials.php

```
<?php
  $db_host = "159.149.84.152";              // The server hosting the database
  $db_user = "crestaz";                     // Our username for the database server
  $db_pass = "***********";                 // The password for our account
  $db_db = "";                      // The database name for our account (unused)
?>
```

tstype.php

```
<?php
/*
 * Author: E. Crestaz
 * Date:   December 7th, 2010
 * Scope:  Access spatial database and send to browser
 */
header ("content-type: text/xml");
?>
<?php
  // Connect to the PostGIS database
  require('db_credentials.php');
  $db_handle = pg_connect("host=$db_host port=5434 dbname=crestaz user=$db_user
password=$db_pass");

  // It must not print anything else other than points
  /*
  if ($db_handle) {
    //echo 'Connection attempt succeeded!';
  }
  else {
    // It should be commented ...!
    echo 'Connection attempt failed!';
  }
  */

  $query = "SELECT *
            FROM ems.tstype";

  $result = pg_query($db_handle, $query);
?>
<?php
  // Coded based on http://www.tonymarston.co.uk/php-mysql/dom.html
  // Create a new XML document
  $doc = new DomDocument('1.0');  //,'iso-8859-1');

  // See Pamela Fox example at
  // http://gmaps-samples-v3.googlecode.com/svn/trunk/xmlparsing/moredata.xml

  // Set a nice output
  $doc->formatOutput = true;

  // Create root node
  $root = $doc->createElement('tstypes');
  $doc->appendChild($root);

  // Process one row at a time
  for ($row=0; $row<pg_num_rows($result); $row++) {
        $values = pg_fetch_object($result,$row);
        // Refer to Hydro Data Model specifications for semantics
    $id = $values->id;                // Measurement type id
```

```
50      $code = $values->code;            // Measurement code (i.e. h, CHg)
51      $variable = $values->variable;    // Measurement type long description (i.e. Piezometric
head)
52          $units = $values->units;          // Measurement units (i.e. m a.s.l.)
53          $datatype = $values->datatype;    // Measurement datatype (i.e. instantaneous)
54      $origin = $values->origin;        // Measurement origin (i.e. observed, computed)
55
56      // Create child element
57          $tstype = $doc->createElement("tstype");
58      $root->appendChild($tstype);
59
60      // Create text node
61          /*
62          $text = $doc->createTextNode("text value");
63      $marker->appendChild($text);
64      */
65
66      // Create attribute nodes and related values
67      $attr = $doc->createAttribute("id");
68      $tstype->appendChild($attr);
69      $attrValue = $doc->createTextNode("$id");
70      $attr->appendChild($attrValue);
71
72      $attr = $doc->createAttribute("code");
73      $tstype->appendChild($attr);
74      $attrValue = $doc->createTextNode("$code");
75      $attr->appendChild($attrValue);
76
77      $attr = $doc->createAttribute("variable");
78      $tstype->appendChild($attr);
79      $attrValue = $doc->createTextNode("$variable");
80      $attr->appendChild($attrValue);
81
82      $attr = $doc->createAttribute("units");
83      $tstype->appendChild($attr);
84      $attrValue = $doc->createTextNode("$units");
85      $attr->appendChild($attrValue);
86
87      $attr = $doc->createAttribute("datatype");
88      $tstype->appendChild($attr);
89      $attrValue = $doc->createTextNode("$datatype");
90      $attr->appendChild($attrValue);
91
92      $attr = $doc->createAttribute("origin");
93      $tstype->appendChild($attr);
94      $attrValue = $doc->createTextNode("$origin");
95      $attr->appendChild($attrValue);
96    }
97
98    $xml_string = $doc->saveXML();
99    echo $xml_string;
100
101   pg_close($db_handle);
102 ?>
```

gmpg.php

```
<?php
/*
 * Author: E. Crestaz
 * Date:   December 7th, 2010
 * Scope:  Access spatial database and send to browser
 */
header ("content-type: text/xml");

// Connect to the PostGIS database
require('db_credentials.php');
$db_handle = pg_connect("host=$db_host port=5434 dbname=crestaz user=$db_user
password=$db_pass");

// It must not print anything else other than points
  /*
  if ($db_handle) {
    //echo 'Connection attempt succeeded!';
  }
  else {
    // It should be commented ...!
    echo 'Connection attempt failed!';
  }
  */

//$conn = mysql_connect($db_host, $db_user, $db_pass);
//mysql_select_db($db_db, $conn);

$query = "SELECT DISTINCT ON (p.code)
                  p.code,
                  ST_X(p.geom) as lon,
                  ST_Y(p.geom) as lat,
                  ts.tstypeid
            FROM ems.point p, ems.ts ts, ems.tstype tstype
            WHERE p.id = ts.gid AND tstype.id = ts.tstypeid";

  /*
  // Select all points
  $query = "SELECT p.code,
                  ST_X(p.geom) as lon,
                  ST_Y(p.geom) as lat
            FROM ems.point p";
  */

  /*
  $query = "SELECT DISTINCTROW Code, Longitude, Latitude, TsTypeId
              FROM monitoringpoint, ts, tstype
                    WHERE monitoringpoint.ID = ts.monitoringPointID AND tstype.ID =
ts.TsTypeID";
  */

// This part of the program should be reintroduced ...!
//if (isset($_POST["MeasurementType"]) && $_POST["MeasurementType"]!="")
if (isset($_GET["tstypeid"]) && $_GET["tstypeid"]!="") {
//print("tsTypeId set to " . $_GET["tstypeid"]);
//Append at the end of the query a filtering WHERE clause
    $query .= " AND tstype.id = " . $_GET["tstypeid"];
}
else {
//print("tsTypeId not set!");
//$query .= " and tstype.ID = 1";   //-1";   // ??????
}

$query .= " ORDER BY p.code";

// Do not print. Only markers array must be output
// echo ">>>>> " . $query. "<br>";
```

```
65
66 $result = pg_query($db_handle, $query);
67 $joiner = '';
68 $count = 0;
69 ?>
70 <?php
71 /*
72   function parseToXML($htmlStr) {
73     $xmlStr=str_replace('<','&lt;',$htmlStr);
74         $xmlStr=str_replace('>','&gt;',$xmlStr);
75         $xmlStr=str_replace('"','&quot;',$xmlStr);
76         $xmlStr=str_replace("'","&#39;",$xmlStr);
77         $xmlStr=str_replace("&","&amp;",$xmlStr);
78         return $xmlStr;
79   }
80 */
81 ?>
82 <?php
83 // Written based on http://www.tonymarston.co.uk/php-mysql/dom.html
84 // Create a new XML document
85 $doc = new DomDocument('1.0');  //,'iso-8859-1');
86
87 // See Pamela Fox example at
88 // http://gmaps-samples-v3.googlecode.com/svn/trunk/xmlparsing/moredata.xml
89 // <?xml version="1.0" encoding="UTF-8"
90
91 // Display document in browser as plain text
92 // for readability purposes
93 //header("Content-Type: text/plain");
94
95 // we want a nice output
96 $doc->formatOutput = true;
97
98 // Create root node
99 $root = $doc->createElement('markers');
100 $doc->appendChild($root);
101
102 // Process one row at a time
103 for ($row=0; $row<pg_num_rows($result); $row++) {
104     $values = pg_fetch_object($result,$row);
105     $lat = $values->lat;
106     $lon = $values->lon;
107     $code = $values->code;
108
109     // Create child element
110     $marker = $doc->createElement("marker");
111     $root->appendChild($marker);
112
113     // Create text node
114         /*
115         $text = $doc->createTextNode("text value");
116     $marker->appendChild($text);
117     */
118
119     // create attribute nodes and related values
120     $attr = $doc->createAttribute("code");
121     $marker->appendChild($attr);
122     $attrValue = $doc->createTextNode("$code");
123     $attr->appendChild($attrValue);
124
125     $attr = $doc->createAttribute("lat");
126     $marker->appendChild($attr);
127     $attrValue = $doc->createTextNode("$lat");
128     $attr->appendChild($attrValue);
129
130     $attr = $doc->createAttribute("lon");
131     $marker->appendChild($attr);
132     $attrValue = $doc->createTextNode("$lon");
133     $attr->appendChild($attrValue);
134
```

```
/*
    $attr = $doc->createAttribute("lat");
    $marker->appendChild($attr);
    $attrValue = $doc->createAttribute($lat);
    $marker->appendChild($attrValue);

    $attr = $doc->createAttribute("lon");
    $marker->appendChild($attr);
    $attrValue = $doc->createAttribute($lon);
    $marker->appendChild($attrValue);
*/
}

$xml_string = $doc->saveXML();
//echo "Test from E. Crestaz";
echo $xml_string;

// End XML file
pg_close($db_handle);
?>
```

timeseries.php

```
<?php
header ("content-type: text/xml");
/*
* Author: Ezio Crestaz
* Date:   December 2010
* Scope:  Build a XML file with time series related passed parameters, namely point code and
*         time series type, and return to the browser
*/
$code = $_GET["code"];
$tstypeid = $_GET["tstypeid"];
// Connect to the PostGIS database
require('db_credentials.php');
$db_handle = pg_connect("host=$db_host port=5434 dbname=crestaz user=$db_user
password=$db_pass");
// It must not print anything else other than points
/*
if ($db_handle) {
    //echo 'Connection attempt succeeded!';
}
else {
   // It should be commented ...!
   echo 'Connection attempt failed!';
}
*/

  $query = "SELECT ts.rdate AS rdate,
                   ts.mdate AS mdate,
                                ts.m AS m
            FROM ems.point p, ems.ts ts, ems.tstype tstype
            WHERE p.id = ts.gid AND tstype.id = ts.tstypeid ";

  $query .= "AND p.code=" . '\'' . $code . '\' ';
  $query .= "AND tstype.id=" . $tstypeid;

  $result = pg_query($db_handle, $query);
  $joiner = '';
  $count = 0;
?>
<?php
  // Written based on http://www.tonymarston.co.uk/php-mysql/dom.html
  // Create a new XML document
  $doc = new DomDocument('1.0');  //,'iso-8859-1');

  // See Pamela Fox example at
  // http://gmaps-samples-v3.googlecode.com/svn/trunk/xmlparsing/moredata.xml
```

```
45    // <?xml version="1.0" encoding="UTF-8"
46
47    $doc->formatOutput = true;
48
49    // Create root node
50    $root = $doc->createElement('timeseries');
51    $doc->appendChild($root);
52
53    // Process one row at a time
54    for ($row=0; $row<pg_num_rows($result); $row++) {
55            $values = pg_fetch_object($result,$row);
56      $mdate = $values->mdate;    // Measurement date
57      $m = $values->m;            // Measurement value
58
59      // Create child element
60            $measure = $doc->createElement("measure");
61      $root->appendChild($measure);
62
63      // create attribute nodes and related values
64      $attr = $doc->createAttribute("mdate");
65      $measure->appendChild($attr);
66      $attrValue = $doc->createTextNode("$mdate");
67      $attr->appendChild($attrValue);
68
69      $attr = $doc->createAttribute("value");
70      $measure->appendChild($attr);
71      $attrValue = $doc->createTextNode("$m");
72      $attr->appendChild($attrValue);
73    }
74
75    $xml_string = $doc->saveXML();
76
77    echo $xml_string;
78
79    // End XML file
80    pg_close($db_handle);
81  ?>
```

APPENDIX 5 – VISUAL BASIC CODE FOR CREATION OF MULTIPLE SQL INSERT STATEMENTS ON TIME SERIES TABLE

createInsertStatements

```
Option Explicit

'
' Author: E. Crestaz
' Date:   13.11.2010
' Scope:  read in a CSV text file and output a SQL text file to be used
' for data insertion to a PostGIS table. While PostGIS provides both an SQL COPY ' statement
(referring to a text file on server) and a \copy plsql command
' (working with a text file on local disk), both options do not make use of
' any sequence which has been eventually defined. So, just to provide an
' example, if a PK id has been defined as serial for explicit generation, it
' should anyway be explicitly defined in text file. If at a later stage,
' other records should be added using INSERT INTO SQL statements, conflicts
' would arise, due to key duplication problems. The adoption of a sequence
' starting value higher than the maximum inserted value could provide a
' solution.
'
Sub createInsertStatements()
  Dim dirIn As String    ' Input and output directories and file names
  Dim dirOut As String
  Dim fileIn As String
  Dim fileOut As String
  Dim gid As Long        ' FK referencing geometry
  Dim tstypeid As Long   ' FK referencing time series type description
  Dim mdate As String    ' Measurement date as a string
  Dim m As Double        ' Measurement value
  Dim n As Long          ' Number of records
  Dim q As String        ' Simple quote

  q = Chr$(39)

  dirIn =
"C:\Ezio\Formazione\Hertfordshire\MScComputerScience\Courses\MScDissertation\Application\Data\Cas
eStudy\txt"
  dirOut =
"C:\Ezio\Formazione\Hertfordshire\MScComputerScience\Courses\MScDissertation\Application\SourceCo
de\SQL"
  fileIn = "tmpts.txt"
  fileOut = "insert_tmpts.sql"

  Open dirIn + "\" + fileIn For Input As #1
  Open dirOut + "\" + fileOut For Output As #2

  Print #2, "INSERT INTO ems.ts (gid,tstypeid,mdate,m)"
  Print #2, "VALUES"

  n = 0
  While Not EOF(1)
    Input #1, gid, tstypeid, mdate, m

    If n > 0 Then Print #2, ","  ' Closing record comma
    'Print #2, "  (" + Str$(gid) + "," + Str$(tstypeid) + ",";
    'Print #2, "to_date(" + q + mdate + q + "," + q + "dd/mm/yyyy" + q + "),";
    'Print #2, Str$(m) + ")";
    Print #2, "  (" + Trim$(Str$(gid)) + "," + Trim$(Str$(tstypeid)) + ",";
    Print #2, "to_date(" + q + mdate + q + "," + q + "dd/mm/yyyy" + q + "),";
    Print #2, Trim$(Str$(m)) + ")";
    n = n + 1
  Wend
```

```
55    Print #2, ""      ' Move to next line
56    Print #2, ";"    ' Closing VALUES brace
57
58    Close #1
59    Close #2
60
61 End Sub
```