DOCUMENTO PROVISÓRIO

Carlos Manuel
Basílio Oliveira

Arquitectura de Software Escalável para Sistemas de Apoio à Decisão para Entidades Gestoras de Àgua

Towards a scalable Software Architecture for Water Utilities' Decision Support Systems

DOCUMENTO PROVISÓRIO

Carlos Manuel Basílio Oliveira

Arquitectura de Software Escalável para Sistemas de Apoio à Decisão para Entidades Gestoras de Àgua

Towards a scalable Software Architecture for Water Utilities' Decision Support Systems

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor André Zúquete, auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor António Gil D'Orey Andrade Campos (co-orientador), Professor auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro.

o júri / the jury

presidente / president

ABC
Professor Catedrático da Universidade de Aveiro (por delegação da Reitora da Universidade de Aveiro)

vogais / examiners committee

DEF
Professor Catedrático da Universidade de Aveiro (orientador)

GHI
Professor associado da Universidade J (co-orientador)

KLM
Professor Catedrático da Universidade N

agradecimentos

Agradeço o apoio da minha família, amigos e colegas da SCUBIC, e ao prof. Zúquete pela paciência e disponibilidade estes últimos anos

acknowledgments

I wish to thank my family, friends and coworkers at SCUBIC for the support, as well as prof. Zúquete for the availability and patience through these past years

Palavras-chave

Àgua, Arquitectura de Software, Sistemas de Apoio à Decisão, Entidades Gestoras de Água

Resumo

O fornecimento de àgua às populações é um símbolo de qualquer grande sociedade, desde o início da Civilização. Hoje em dia, enormes quantidades de àgua são fornecidas constantemente a residências e indústrias variadas utilizando motores eléctricos acopolados a bombas de àgua que consomem vastas quantidades de energia eléctrica. Com o recurso a tarifas de electricidade variáveis e dinâmicas, dados em tempo real de variados sensores nas empresas de fornecimento de àgua e a modelos da rede de distribuição de àgua, o software da SCUBIC consegue monitorizar e prever consumos de àgua e assim optimizar a operação destas bombas por forma a baixar os custos operacionais das empresas gestoras de àgua.

O software fornecido pela SCUBIC é uma amálgama de serviços construídos numa fase embrionária da empresa, que por se manterem inalterados ao longo dos anos, não se adequam mais ao plano de negócios e escalada de requisitos por parte dos *stakeholders*. Daqui surge então a necessidade de construir uma nova arquitectura de software capaz de responder aos novos desafios numa indústria cada vez mais instrumentalizada e evoluída como a da Gestão de Água.

Recorrendo a métodos de engenharia de software, migração de arquitecturas de software e planeamento cuidadoso, foi possível alterar a arquitectura do software usado pela SCUBIC. Após rever os resultados gerados pelos indicadores de performance, conclui-se que a migração foi um sucesso.

Keywords Key, word.

Abstract

Water Supply is a staple of all civilizations throughout History. Nowadays, huge amounts of water are constantly supplied to homes and businesses, requiring the use of electric pumps which consume vast amounts of electric energy.
By using variable and dynamic electric tariffs, multiple real-time sensor date from Water Utilities and Water Network Modelling, the SCUBIC software is able to monitor the water networks, predict water consumption and optimize pump operation allowing the Water Utilities to lower operational costs.
Built during an earlier phase of the company, the SCUBIC software is a monolithic amalgamation of services, full of compromises that cannot fulfill the latest requirements from the *stakeholders* and business plan.
Therefore, a need to build a more modular and scalable software architecture for this software becomes apparent. Using careful planning, software engineering knowledge and literature regarding software architecture migration, a new software architecture was implemented. Results from comparisons between the older and newer architectures prove that the migration was a success and complies with the requirements set at the beginning of the project.

Table of contents

List of figures

List of tables

List of abbreviations

AFK . . . . . away from keyboard
AWS . . . . . Amazon Web Services

DSS . . . . . Decision Support System

EC2 . . . . . Elastic Compute Cloud
EIP . . . . . Elastic IP

GW . . . . . good work
GWHF . . . . . GW and HF

HF . . . . . have fun

MSc . . . . . Master of Science

NIC . . . . . Network Interface Card

PDF . . . . . portable document format
PhD . . . . . Doctor of Philosophy

SCADA . . . . . Supervisory Control And Data Acquisition

VPC . . . . . Virtual Private Cloud
VPS . . . . . Virtual Private Server
VSD . . . . . Variable-Frequency Drive

WSS . . . . . Water Supply Systems
WU . . . . . Water Utilities

Chapter 1

Introduction

This chapter presents the overall theme of this body of work. Firstly, some context is given about the overall theme of this body of work and the motivation behind it. Then, the objectives for dissertation are presented to the reader. Finally, at the end of the chapter, some information regarding the content of each chapter is presented.

1.1 Water Supply Systems

The water supply systems that are prevalent in our society play a very important role in our daily lives, distributing water throughout the country from water reservoirs or water treatment plants up until it reaches our houses and industries. These Water Supply Systems (WSS) can be quite complex and difficult to manage without proper processes that ensure the operation of such networks is made without problems, in an environmental and economically sustainable way. For this reason, the use of specialized software to aid operators or even automatically control the operation of these WSS is of uttermost importance nowadays. Water, be it in quantity and quality, has been a staple of all major human civilizations throughout History, from ancient roman aqueducts to the current era.

Moving large quantities of water through enormous WSS requires the use of vast quantities of mechanical work, which in turn requires lots of energy, namely, electric energy. With the ever-growing political, economic and environmental pressure to improve and optimize how we use energy, and with the current geopolitical issues, access to energy is getting more expensive and regulated. This means that the need for the optimization of pumping operations to reduce costs and, potentially lower energy use as well, is growing within Water Utilities (WU).

1.2 Existing Decision Support System

In order for the WU's to optimally operate their water pumps, a Decision Support System (DSS) is used by the WU's pump operators and/or by automatic Supervisory

Control And Data Acquisition (SCADA) systems. This DSS is a web platform designed to suggest *which* pumps to operate, *when* to operate, for *how* long to operate and in some cases what *speed* their Variable-Frequency Drive (VSD)'s should operate at like shown in Figure 1.1

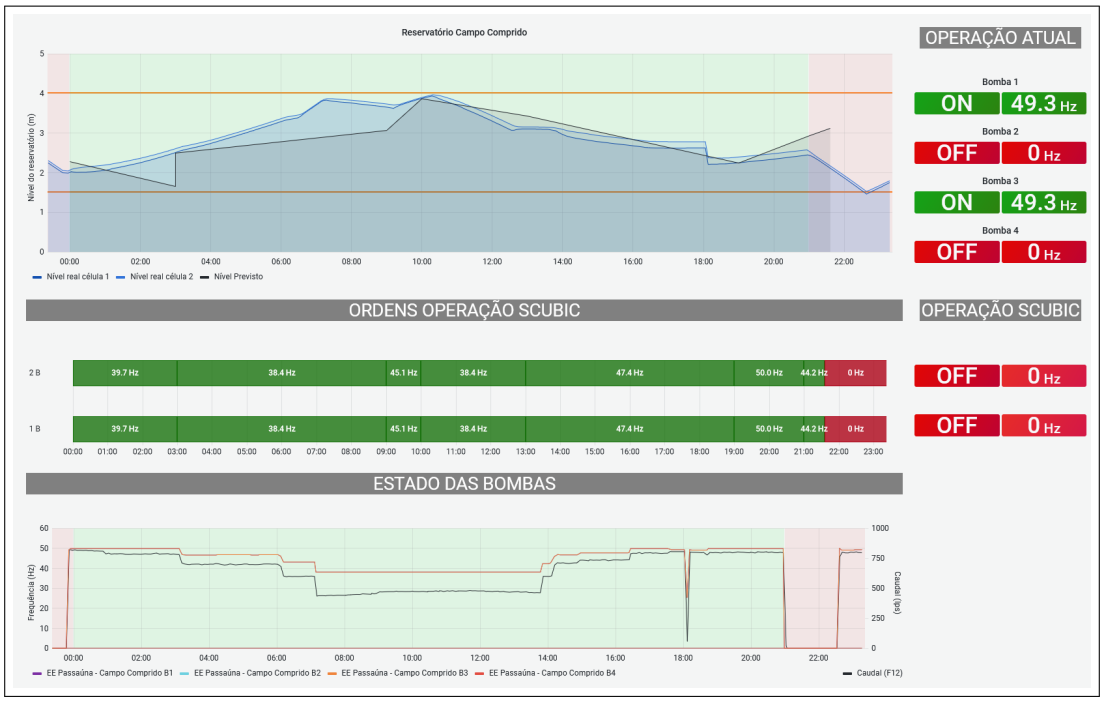

Figure 1.1: Example of a DSS interface that uses *Grafana*.

The existing software's architecture can be summarized as a "Monolithic Modular" software architecture (Newman, 2019). This architecture is composed of a set of Virtual Private Server (VPS), one for each Client, where a set of Docker containers contain (containers contain?) all the services needed for running the software for that Client. These services are also configured and developed separately, each in a different code repository and as such there is an unsurmountable amount of code drift between the same services of the different clients. This is, quite apparently, not scalable nor even remotely manageable for any software development team. Further ahead, on Section 3.1.1, a complete analysis of this architecture will be provided and explained in detail.

1.3 Objectives

The main goal of this work is to make the migration from the old software architecture of the DSS to a better, improved software, considering the requirements from the stakeholders. This new architecture hopes to improve the performance, reliability, resilience, security and scalability of the old DSS Architecture. A new, better architecture software brings improvements not just for the software itself but also for the development team, allowing them to improve and maintain the software easier and faster than ever before.

By reducing the amount of work and time the software development team spends on each maintenance action or new functionality, it's also hoped to save monetary resources to the software company as well. Infrastructure costs are also an important aspect of this new architecture, where the adoption of more modular and independent services might mean a more optimal use of compute resources, resulting in lowering such costs.

1.4 Structure of the Document

This document is composed by a total of X chapters.

In Chapter 1, the reader is presented with a summary of the work to be done as well as the context behind such decision.

In Chapter 2, a bibliographical analysis regarding the state of software architecture and cloud-based software solutions is presented. It's divided in two sections, starting with some insight into how a Software Architecture is planned, executed and then analyzed. Some text regarding the general technologies used throughout the work is also analyzed here.

Chapter 3 is divided into multiple sections. Firstly, a more detailed explanation of the old architecture and its inherent flaws is presented, flaws which end up showcasing the need for a new and improved software architecture. The second section is related to the first step when engaging a new engineering project: Requirements. In this section, the goals for the new architecture are laid out along the multiple constraints that are in place throughout the whole execution of the work. Here, the methodology to be adopted for this work is presented as well. In a third section, the plans for implementing the new architecture are laid out, chronologically. There wasn't a single plan because the requirements and constraints kept changing during the planning and implementation part of the work. Lastly, the final section is related to the actual implementation of the proposed software architecture. The procedures taken, the challenges and decisions made throughout the implementation are shown and contextualized in this section. In here, the finalized architecture is shown with the help of diagrams. (Where can/should I introduce the methodology for measuring the performance indexes and other benchmarks that analyze the new architecture?)

Chapter 4 is where we can see whether the new architecture complies with the restraints imposed by the stakeholders, achieves the required and desired results and how it compares with the old architecture. In a first part, we analyze the functional requirements, followed by the non-functional requirements and overall feedback from the development team that accompanied this software architecture migration. Finally, a cost analysis is made for both the recurring monetary infrastructure costs and overall impact on team productivity.

Chapter 5 discusses the previous results and presents some conclusions from what has been demonstrated on previous chapters.

Furthermore, attached to this document, is an appendix that contains some extra results generated from the monitoring interface used internally to evaluate the new archi-

tecture.

Chapter 2

State-of-the-Art

2.1 Software Engineering

2.1.1 Defining Requirements

Here, we will show what's the state-of-the-art regarding requirements definition. This is an important step in Software Engineering, or any Engineering.

2.2 Software Architecture

2.3 Cloud-Based

Chapter 3

Methodology

3.1 Requirements

3.1.1 Current Architecture

The current software architecture is still in use as of the date of publication of this body of work. This older architecture consists of an amalgamation of Docker containers, each running a different service. Each Client has its own VPS wherein these Docker containers are deployed.

Virtual Private Cloud (VPC)

These VPS are general-purpose Amazon Web Services (AWS) Elastic Compute Cloud (EC2) *Instances*. As can be seen on the diagram presented on Figure 3.1, these instances are deployed to the same VPC, sharing a private network between them. The Reverse Proxy serves as, as the name implies, as a reverse proxy to enable the use of a single Elastic IP (EIP), a single Network Interface Card (NIC) by all Clients's servers, since the availability of public IPs is limited to five EIP.

Figure 3.1: The AWS VPC used, hosting the old architecture's EC2 VPS

VPC

Each EC2 instance runs a Docker container for each one of the following services:

- **InfluxDB** (Timeseries Database)
- **MongoDB** (General use, no-SQL, Document Database)

- **Grafana** (Web platform for data visualization, the front end of the DSS)
- **Telegraf** (Data collecting service)
- **Nginx** (Reverse proxy with HTTPS capabilities)
- **Let's Encrypt** (Automatic SSL Certificate emitter, companion for the Nginx container)
- **Web Dev** (Web platform / API for managing Workers' settings)
- **Redis** (Message Queue System for queuing Worker's jobs)
- **OpenSSH** (*atmoz/sftp*) (SFTP Server for receiving client data)
- **Workers** (Container running the Forecast, Simulation and Optimization Python Algorithms as well as the KPI Algorithms.)
- **Workers** (*Beat*) (Container that periodically *triggers* jobs in the Workers container)

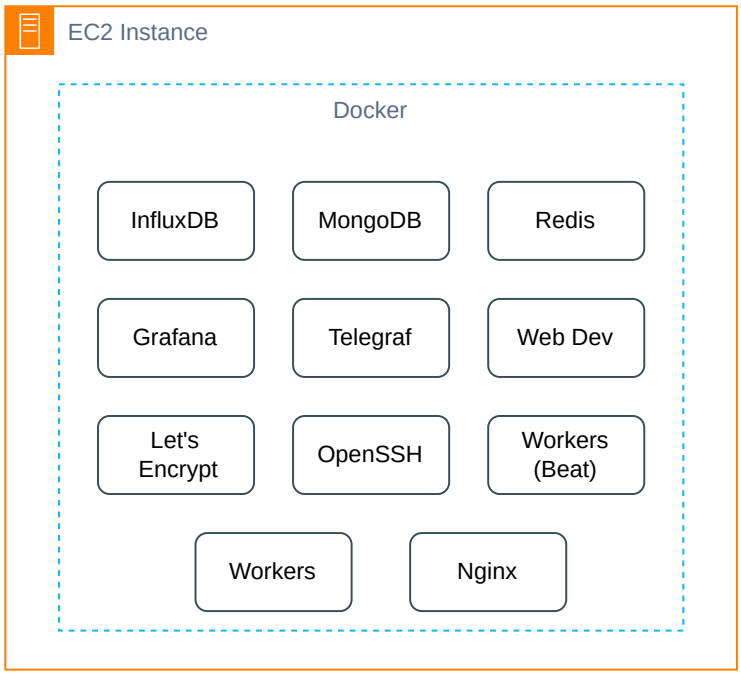

Figure 3.2: A singular AWS EC2 instance, hosting the old architecture's docker containers

Databases

There are two types of databases being used by this architecture: A Timeseries Database, in this case **InfluxDB**, and an additional general-purpose Document Database: **MongoDB**. Each type of database has a different role, the first one stores the Client's timeseries data such as sensor information, pump orders, predicted tank levels, etc. The

second one, the Document Database, is responsible for storing configuration settings for each worker service (optimization, simulation and forecasting), for storing electrical tariffs data and to store sensor device's configurations.

Grafana

This web platform allows the visualization of the Timeseries data from the **InfluxDB** database. This is a freely-available platform that runs on a docker container with little to no modifications necessary. The dashboards are built using the built-in tools and allow for complex and very informative data visualization. This is used in both the new and old architecture, since the new visualization platform is still not operational (not within the scope of this body of work).

Telegraf

The **Telegraf** container is used to gather the files containing the raw sensor data sent from the Client to the **SFTP** server. Since this container shares the file upload location folder with the **SFTP**, through a convoluted process of storing the filename of the last file uploaded, periodically checking for the next file and file handling *spaghetti* code that spans multiple files and has an enormous codebase that weighs the docker image's file size considerably.

Chapter 4

Results and Discussion

Chapter 5

Conclusion

Chapter 6

The package

This chapter presents briefly how to use the `uathesis` package.

6.1 UA thesis LaTeX style file

The `uathesis` LaTeX style file was originally created by professor Tomás Oliveira e Silva (2012), and it is currently available at his home page[1]. In this template it is used a new version modified by Rui Antunes[2].

6.1.1 Options

The following options are supported:

- `oldLogo`: to use the old logo of University of Aveiro.
- `newLogo`: to use the new logo of University of Aveiro (default behavior).
- `MAP`: for MAP joint doctoral programmes. The logos from the three universities (Aveiro, Minho, Porto) are used.
- `draft`: it prints "DOCUMENTO PROVISÓRIO" in the first two front pages.
- `draftPT`: same as `draft`.
- `draftEN`: same as `draft`, but instead it prints "DRAFT DOCUMENT".
- As of May 29, 2021, the department name shall not appear in the cover and the first page (top header). A new option, `NODEPT` (no department), was created to suppress the department name (now this is the default behavior).
  However, formerly the department name would appear in the cover and first page, therefore the old options were kept for the sake of preservation. Any department

[1] http://sweet.ua.pt/tos/TeX.html
[2] https://github.com/ruiantunes/ua-thesis-template

name can be shown by using one of the following options: `DAO`, `DBIO`, `DCM`, `DCSPT`, `DECA`, `DECIVIL`, `DEGEIT`, `DEM`, `DEMAC`, `DEP`, `DETI`, `DFIS`, `DGEO`, `DLC`, `DMAT`, `DQ`.

- The color of the top bar, in the cover page, is defined by specifying one of the following scientific areas: `accounting`, `arts`, `economy`, `education`, `engineering`, `health`, `humanities`, `sciences`.

Chapter 7

Tips and examples

This chapter presents some basic tips and a few examples on how to use LaTeX.

7.1 TeX engines

There are several TeX engines. In short, these are used to compile the (La)TeX source code to generate the output file (for example, a PDF). To know more about these, I encourage you to check these articles:

- The TeX family tree: LaTeX, pdfTeX, XeTeX, LuaTeX and ConTeXt.
  https://www.overleaf.com/learn/latex/Articles/The_TeX_family_tree:_LaTeX,_pdfTeX,_XeTeX,_LuaTeX_and_ConTeXt
- Choosing a LaTeX Compiler.
  https://www.overleaf.com/learn/latex/Choosing_a_LaTeX_Compiler
- Are there benefits to use XeTeX or LuaTeX if one is to write documents mainly in English?
  https://tex.stackexchange.com/questions/548467/are-there-benefits-to-use-xetex-or-luatex-if-one-is-to-write-documents-mainly-in
- Why choose LuaLaTeX over XeLaTeX.
  https://tex.stackexchange.com/questions/126206/why-choose-lualatex-over-xelatex
- Differences between LuaTeX, ConTeXt and XeTeX.
  https://tex.stackexchange.com/questions/36/differences-between-luatex-context-and-xetex

In this template, support for both pdfTeX and LuaTeX engines has been guaranteed, but I encourage you to use the latter because it is more powerful for typefaces: it supports TrueType and OpenType standards.

7.1.1 Compiler automatic detection

pdfTeX is being used.

Consider changing to LuaTeX, which is the recommended compiler for this template.

7.2 Basic tips

- The \\ command has the same effect as `\newline`.
- The `\cleardoublepage` command forces the next content to start in an odd page.
- The tilde character (~) inserts a non-breaking space. Use it before citing a reference to avoid breaking the line: `an example~\cite{label}`.
- The current font size is 10.95pt.
- Use the `longtable` environment for tables spanning multiple pages.
- The grave accent ` and the apostrophe ' are the correct symbols to make quotations: “this is an example”.

7.3 Font styles

- `\textnormal{}` — document font family.
- `\textrm{}` — roman font family.
- `\textsf{}` — sans serif font family.
- `\texttt{}` — `teletypefont family`.
- `\textit{}` — *italic shape*.
- `\textsl{}` — *slanted shape*.
- `\textsc{}` — small capitals.
- `\textbf{}` — **bold**.

7.4 Colors

Red colored text from the `color` package. And Blue4 colored text from the `xcolor` package.

7.5 Footnotes

This is a labeled footnote[1]. A footnote can be referenced multiple times[1]. Again, the same footnote is referenced[1].

7.5.1 A table example

A table example is shown in Table 7.1.

Table 7.1: A table example.

justified a bc def ghij klm no p qr stu vwxy z	left-aligned a bc def ghij klm no p qr stu vwxy z	centered a bc def ghij klm no p qr stu vwxy z	right-aligned a bc def ghij klm no p qr stu vwxy z
This is an example	2	3	4
Single cell	A multi-column cell		
A multi-row cell	A simple example	... ...	... ...

7.6 Abbreviations

`\gls{label}` and `\glslink{label}{text}` are two possible commands for making use of abbreviations. For example, the commands `\gls{afk}` (first call), `\gls{afk}` (second call), and `\glslink{afk}{insert specific text}` produce respectively "away from keyboard (AFK)", "AFK" and "insert specific text".

A list of some commands follow.

- `\gls{afk}` produces "AFK".
- `\glslink{afk}{text}` produces "text".
- `\glsxtrshort{afk}` and `\as{afk}` produce "AFK" and "AFK", respectively.
- `\glsxtrlong{afk}` and `\al{afk}` produce "away from keyboard" and "away from keyboard", respectively.

Note that the commands `\as{}` and `\al{}` are shorter variants.

Other abbreviations include: good work (GW); have fun (HF); good work and have fun (GWHF). Note that the latter contains nested abbreviations.

[1] This is a footnote example.

7.7 Equations

Equation (7.1) is a numbered equation.

$$x = 1 + y \tag{7.1}$$

The following equation is not numbered, and thus cannot be referenced.

$$y = \sum_{i=1}^{N} x_i$$

The `\myscore` pre-defined math function is used by the Equation (7.2).

$$\mathrm{score}(d) = \frac{1}{d^2} \tag{7.2}$$

7.8 Figures

An example of a figure is shown in Figure 7.1. The `\fbox` command draws a box around its content.

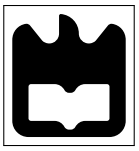

Figure 7.1: This caption is shown below the figure.

7.9 Rotating pages

Horizontal pages can be obtained using the `landscape` environment from the `pdflscape` package:

```
\begin{landscape}
Add some text or figures.
\end{landscape}
```

7.10 A long table

Table 7.2 presents a long table using the `longtable` package. This table can span multiple pages. The `\afterpage` command forces the table to start at the top of a page.

Table 7.2: A long table.

Column 1	**Column 2**
0	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
1	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
2	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
3	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
4	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
5	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
6	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
7	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z

8	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z
9	a bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy za bc def ghij klm no p qr stu vwxy z

Chapter 8

How to reference

This chapter presents how to reference (1) documents such as books, journal articles, conference articles, among other works and (2) elements of this document, such as, chapters, sections, subsections, equations, figures, and tables.

8.1 Citing other works

In this template, it is used the `biblatex` package for citing other works. It is important to note that BibTeX and BibLaTeX have many similarities but are two different formats. For example, if you want to indicate a conference's place you have to use the `address` field in BibTeX, but the `venue` field in BibLaTeX. The `address` field in BibLaTeX is used to indicate the publisher's local. If you would like to indicate the publisher in journal articles I suggest you to use the `note` field. By default, the references are expected to be in the `refs.bib` file. Organize your citations with a reference manager such as, for example, Zotero[1] with the Better BibTeX extension[2]. Zotero is what I personally use and recommend. For my citation key labels I use the format `[author][year][a]`: lowercase string with the last name(s) of the first author followed by the year of the publication and a suffix letter for distinction of repeated works (check the `refs.bib` file for some examples).

8.1.1 The citation style

There are several citation styles. You can configure your own, or use one of the three configurations that are already provided in this template:

- Author-year style:
  `\input{tex/config/biblatex/authoryear.tex}`.
- Author-year compact style:
  `\input{tex/config/biblatex/authoryear-comp.tex}`

[1] https://www.zotero.org/
[2] https://retorque.re/zotero-better-bibtex/

- Numeric style:
  `\input{tex/config/biblatex/numeric.tex}`.

You can choose one of these in the `tex/config/packages.tex` file. By default, the `authoryear-comp` style is selected.

8.1.2 The citation commands

A list of citation commands, and examples, follows.

- `\cite{oliveiraesilva2012a}` produces "Oliveira e Silva, 2012".
- `\cite{oliveiraesilva2012a,oliveiraesilva1994a}` produces "Oliveira e Silva, 2012, 1994".
- `\textcite{oliveiraesilva2012a}` produces "Oliveira e Silva (2012)".
- `\textcite{oliveiraesilva2012a,oliveiraesilva1994a}` produces "Oliveira e Silva (2012, 1994)".
- `\parencite{oliveiraesilva2012a}` produces "(Oliveira e Silva, 2012)".
- `\parencite{oliveiraesilva2012a,oliveiraesilva1994a}` produces "(Oliveira e Silva, 2012, 1994)".
- `\parencite*{oliveiraesilva2012a}` produces "(2012)".
- `\parencite*{oliveiraesilva2012a,oliveiraesilva1994a}` produces "(2012, 1994)".
- `\fullcite{oliveiraesilva2012a}` produces:
  "Tomás Oliveira e Silva (May 2012). *UA thesis LaTeX style file.*
  URL: http://sweet.ua.pt/tos/TeX/ua_thesis.tgz".

The `\cite{label}` command is preferred with the numeric style. In the `authoryear` and `authoryear-comp` styles, the `\textcite{label}` and `parencite{label}` commands should be used.

Multiple citations are allowed by using multiple labels within the same command. Example: `\cite{oliveiraesilva2012a,oliveiraesilva1994a}`.

More examples follow. A PhD thesis (Oliveira e Silva, 1994). A MSc dissertation (Antunes, 2015). Conference or workshop articles (Antunes *et al.*, 2019, 2020; Almeida *et al.*, 2021). A journal article (Antunes and Matos, 2019).

8.2 Referencing elements of this document

Chapters 6 and 8 are two chapters.

Chapters 6 to 8 are three consecutive chapters.
Sections 8.1 and 8.3 are two sections.
Equations (7.1) and (7.2) are two numbered equations.
Figure 7.1 is a figure.
Table 7.1 is a table.
Sections 8.3 and 8.3.1 have different levels of depth.
Appendices A and B are two appendices.

8.3 A section

This section presents the different levels of depth.

8.3.1 A subsection

Text in a subsection.

A subsubsection

Text in a subsubsection.

A paragraph
Text in a paragraph.

A subparagraph
Text in a subparagraph.

Chapter 9

Future work

Feedback is welcome. This template may have future improvements.

References

Almeida, Tiago, Rui Antunes, João Figueira Silva, João Rafael Almeida, and Sérgio Matos (Nov. 2021). "Chemical detection and indexing in PubMed full text articles using deep learning and rule-based methods." In: *BioCreative VII Challenge Evaluation Workshop* (Online), pp. 119–123.
URL: https://biocreative.bioinformatics.udel.edu/resources/publications/bc-vii-workshop-proceedings/ (cit. on p. 24).

Antunes, Rui (July 2015). "Genomics music." Master's thesis. University of Aveiro.
URL: http://hdl.handle.net/10773/18866 (cit. on p. 24).

Antunes, Rui and Sérgio Matos (Oct. 2019). "Extraction of chemical–protein interactions from the literature using neural networks and narrow instance representation." In: *Database* 2019.baz095. Oxford University Press.
URL: https://doi.org/10.1093/database/baz095 (cit. on p. 24).

Antunes, Rui, João Figueira Silva, and Sérgio Matos (Mar. 2020). "Evaluating semantic textual similarity in clinical sentences using deep learning and sentence embeddings." In: *35th Annual ACM Symposium on Applied Computing* (Online). ACM, pp. 662–669.
URL: http://hdl.handle.net/10773/31473 (cit. on p. 24).

Antunes, Rui, João Figueira Silva, Arnaldo Pereira, and Sérgio Matos (Feb. 2019). "Rule-based and machine learning hybrid system for patient cohort selection." In: *12th International Joint Conference on Biomedical Engineering Systems and Technologies* (Prague, Czech Republic). SciTePress, pp. 59–67.
URL: https://doi.org/10.5220/0007349300590067 (cit. on p. 24).

Newman, Sam (2019). *Monolith to microservices: evolutionary patterns to transform your monolith*. O'Reilly Media. (Cit. on p. 2).

Oliveira e Silva, Tomás (Apr. 1994). "Sobre os filtros de Kautz e sua utilização na aproximação de sistemas lineares invariantes no tempo." PhD thesis. University of Aveiro.
URL: http://hdl.handle.net/10773/4641 (cit. on p. 24).

Oliveira e Silva, Tomás (May 2012). *UA thesis LaTeX style file*.
URL: http://sweet.ua.pt/tos/TeX/ua_thesis.tgz (cit. on pp. 15, 24).

Appendices

Appendix A

Appendix example

This is the first appendix.

A.1 A section example

Similarly to a chapter we can add sections, subsections, and so on...

Appendix B

A second example of an appendix

This is the second appendix.