APIs in Digital Infrastructures of the Greek Government

Michail Loukeris

Supervisor: Prof. P. Louridas

Department of Management Science and Technology
Athens University of Economics and Business

This dissertation is submitted for the degree of
Management Science and Technology

December 2019

I would like to dedicate this thesis to my loving parents . . .

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Michail Loukeris
December 2019

Acknowledgements

And I would like to acknowledge ...

First and foremost, my academic supervisor, Prof. Panagiotis Louridas for supporting me during this semester by pointing me to the right direction in choosing an interesting, both academically and practically, topic for this thesis, as well as for his immediate support whenever I needed it. As well as Prof. Diomidis Spinellis for teaching me what a good research paper looks like.

I would also like to show gratitude to all my colleagues in GRNet, who treated me first as a friend, and taught me all the know when it comes to recruiting and selecting employees. Moreover, I would like to point out Ntora Morfi and George Korfiatis from GRnet and thank them for trusting me and bringing me onboard to their organization.

Finally, I would like to thank my family and friends. Without their patience, encouragement and support all these four years of my studies, this paper would have never been accomplished.

Abstract

New technological developments have strongly influenced the way we live our everyday lives, the way we perform daily activities such as shopping or communication. Every day, more and more people have access to the Internet and to newest technological tools which help them make their lives easier and of better quality. This is something that since the past twenty years Governments of different countries have tried to take advantage of, by applying all these new tools to their different functions in order to simplify their services, make them globally accessible and reduce the red tape and generally become more effective both internally as well as externally.

Countries such as Estonia, the UK, Singapore or Denmark have managed to migrate to a fully digital government where every operation such as tax paying, transportation and drive license issuing, employing people and more are done digitally.

This study provides an analysis of Web APIs as enablers for the digital transformation of government. While digital transformation of government is much wider than the technologies which can potentially support it, an analysis of the role of Web APIs in the public sector is highly relevant to illustrate how technology can enable the transformation of government. The aim of this work has been to identify the ability of Web APIs to assist Member States with enabling their digital transformation. Areas of specific focus include cross-border interoperability between Member States and the opportunity for the EU to become involved in developing or advocating API standards.

This research set out to explore the API landscape in the EU public sector. API is the acronym for Application Programming Interface and it refers to a set of clearly defined methods of communication between a service and any other software or components,essentially, a software intermediary that allows two applications to interact with each other. The purpose of the study has been to identify the ability of Web APIs (hereafter “APIs”) to assist Member States with enabling their digital transformation. Areas of specific focus include aspects such as cross-border interoperability between Member States, and the opportunity for the EU to become involved in developing or advocating API standards.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

At a policy level, the Tallinn Declaration, signed on 6th October 20172 , confirms the commitment to the vision laid out in the EU eGovernment Action Plan 2016-20203 and in the European Interoperability Framework (EIF)4 . In the next five years (2018-2022), steps will be taken towards the declared principles in EU public administrations, namely: "digital-by-default, inclusiveness and accessibility", "once-only", "trustworthiness and security", "openness and transparency", and "interoperability by default", as well as national interoperability frameworks based on the European Interoperability Framework (EIF). In the Declaration the "user-centricity principles for design and delivery of digital public services" is key. When interacting with public administrations and using digital public services, citizens and businesses should have: digital interaction, accessibility, security, availability and usability, reduction of the administrative burden, digital delivery of public services, citizen engagement, incentives for digital service use, protection of personal data and privacy, redress and complaint mechanisms. Whilst not specifically covered within the scope of this study, AI has the potential to improve public services and contribute to the objectives set out in the Tallinn Declaration. For example, the Commission will look into AI's potential to analyse large amounts of data and help check how single market rules are applied5 . Moreover, the digital transformation of society, business and government is raising issues for a range of policy matters across the European Union. As e-government has been in place for the last 20 years, it is timely to explore the interplay between technology and government activities from the perspective of digital government. To understand the intertwined forces that play a role in this transformation process, and their dynamics, contributions from disparate fields and discourses on this topic should be contrasted and compared. At the same time, the Communication on "Building the data economy" (COM (2017) 9) looks at proven or potential blockages to the free movement of data and presents options to remove unjustified and or disproportionate data location restrictions in the EU. It also considers the barriers around access to, and transfer of, non-personal machine-generated data and data liability, as well as issues related to the portability of non-personal data, interoperability and standards. In particular, it calls for the fostering of technical solution development for the reliable identification and exchange of data. A further avenue for research, complementary to this context of the digital transformation of government, is the use of Web Application Programming Interfaces (hereafter called "APIs")). APIs can be seen as "safe entry ports for new and innovative uses of data" held by companies and, potentially, public administrations. An opportunity exists to understand the current context and uptake of this technology early in the innovation cycle of e-government in Europe, including for, but not limited to, geospatial data.

In this context, Gartner was mandated by the Joint Research Centre (JRC) to conduct a study to provide an analysis of the Web APIs as enablers for the digital transformation of governments. While digital transformation of government is much wider than the technologies which can potentially support it, an analysis of the role of APIs in the public sector is highly relevant to illustrate how technology can enable transformation of government. This study explores APIs, and their role in the EU public sector. It is divided up into the following sections: • Introduction. • Methodology: A description of the way in which this study has been conducted using different research designs, including desk based research, consultation of API experts and multiple-case study investigation. Also, using diverse research methods, including API reference and specific document review, and interviews. • Use of APIs in the Public Sector: Explores typical uses of APIs, such as to enable ecosystems, before looking at some specific examples of API use. • Relationship with location data: Explores how adding an additional layer of location data, with other data (via APIs) can provide powerful use cases in many settings such as Smart Cities and Gazetteers. • Differences with the Private Sector: Explores any differences which may exist between the public sector and private sector use of APIs. • Future trends for API use: A view on how APIs may evolve, and their future uses in a world where AI and Robotics are becoming prevalent. • Case Study Insights – Presentation of the findings from case study interviews and associated desk based research, plus analysis. • Conclusions and Recommendations. • Conclusion.

1.1 Glossary

Term	Definition
API	Application Programming Interface - It is a set of clearly defined methods of communication between the service and any other software or components.
API Ecosystem	The developers, and the users of the application constructs they build through an API, either within a company or on the Internet with business partners, customers, citizens etc.
API Economy	A set of business models and channels — based on secure access of functionality and exchange of data to an ecosystem of developers and the users of the app constructs they build — through an API, either within a company or on the Internet with business partners, customers, citizens etc.
API Versioning	The ability to change without rendering older versions of the same API inoperable.
API Standardisation	A uniform way for APIs to be expressed and consumed, from COM and CORBA object brokers to web services to today's RESTful patterns.
API information control	A built-in means for enriching and handling the information embodied by the API. This information includes metadata, approaches to handling batches of records, and hooks for middleware platforms, message brokers, and service buses. It also defines how APIs communicate, route, and manipulate the information being exchanged.

API portal	A means for developers to discover, collaborate, consume, and publish APIs. To support the overall goal of self-service, these portals describe APIs in a way that represents their functionality, context (the business semantics of what they do, and how they do it), non-functional requirements (scalability, security, response times, volume limits, and resiliency dimensions of the service), versioning, and metrics tracking usage, feedback, and performance. For organizations without mature master data or architectural standards, the API portal can still offer visibility into existing APIs and provide contact information for individuals who can describe features, functions, and technical details of services.
API gateway	A mechanism that allows consumers to become authenticated and to “contract” with API specifications and policies that are built into the API itself. Gateways make it possible to decouple the “API proxy”—the node by which consumers logically interact with the service—from the underlying application for which the actual service is being implemented. The gateway layer may offer the means to load balance and throttle API usage.
API brokers	Enrichment, transformation, and validation services to manipulate information coming to/from APIs, as well as tools to embody business rule engines, workflow, and business process orchestration on top of underlying APIs.
API management and monitoring	A centralized and managed control level that provides monitoring, service level management, SDLC process integration, and role-based access management across all three layers above. It includes the ability to instrument and measure API usage, and even capabilities to price and bill charge-back based on API consumption—to internal, or potentially external, parties.

RESTful API	REST stands for “representational state transfer.” APIs built according to REST architectural standards are stateless and offer a simpler alternative to some SOAP standards. For example, REST enables plain-text exchanges of data assets instead of using complex WSDL protocols. It also makes it possible to inherit security policies from an underlying transport mechanism. At a high level, these and other simplified approaches can deliver better performance and faster paths to develop, deploy, and triage.

Table 1.1

1.2 API Overview

APIs have become a foundational technological component of modern digital architectures, impacting every sector of the global economy. In the public sector specifically, APIs are a key enabler of the accelerated evolution of government and its agencies from analogue (manual, paper) operations, to digital. The purpose of our study is to create a report that will support member states with the adoption of APIs when in pursuit of their digital transformation. In order to explore this purpose, our investigation has incorporated (but is not limited to) the following topics: • The current use of APIs in the EU public sector. • Differences between API use in the public sector and the private sector. • The future trends for APIs. • The relationship with location data. • Insights from eight case study interviews, using real world examples to explore different dimensions of the API landscape, including API Strategy, API Ecosystems, and specific APIs. Our approach is covered in detail in the Methodology section later in this report. In summary, we have used a combination of desk based research and interviews to gather information for analysis. For the interviews, we used a questionnaire to collect information from representatives of a set of successful but diverse API based case studies from a range of EU countries and sectors. API Overview API interaction occurs when one application would like to: • Access or query the data held by another application • Send data to that application • Update data held in that application • Request a service from another application Types of APIs Crucially, the use of APIs therefore simplifies, and standardises the interface reducing complexity and cost of deployment over that of custom built interfaces. APIs represent an architectural approach that revolves around providing programmable interfaces to different applications. It is technology agnostic, and creates a flexible, loosely coupled architecture that allows a solution to be made up of components that can more easily be switched in and out. The API approach is also a key enabler for application developers to build apps that rapidly adapt to end user needs. In the public sector, APIs enable important functionality and information held in one agency's system or department to be readily available to another without significant and expensive development effort. As well as cross-departmental (agency) access to functionality and information (or even cross-border with a different country's administration) APIs also provide the ability to share information and functionality more widely, i.e. to developers and ultimately to citizens for consumption through web or mobile based applications. Although there are many different types of API (see Appendix I), this study is most concerned with Web APIs. Web services expose these APIs as endpoints that any internet-enabled language or software can access, in exactly the same way browsers access websites and services7 . Web APIs deliver requests to the service provider, and then deliver the response back to the requestor, i.e. they are an interface for web applications, or applications that need to connect to each other via the Internet to communicate8 . Web APIs themselves can be broken down further based on the type of data format that they harness, for example, well known types are Simple Object Access Protocol (SOAP), Remote Procedure Call (RPC) based APIs, and the Representational State Transfer (REST) architectural style. GraphQL – is a data query language growing in popularity and has been adopted by leading social media outlets such as Facebook and Pinterest9 as a type of API. While typical REST APIs require loading from multiple URLs, GraphQL APIs get all the data an app developer needs in a single request enhancing speed of response even on slow mobile network connections10. Whilst the more traditional APIs are used as integration points within systems hidden from view, Web APIs are often publicly available and can be 'advertised' via API Directory sites online. Tens of thousands11 are available for developers to deliver consumable information to end users to do everything, from checking traffic and weather, to updating a social media status, or even to make payments. In the geospatial domain, besides existing private companies famous API proposals (e.g. Google Map), the Open Geospatial Consortium (OGC) has created standards to support the exchange of geospatial information12. They describe their Web services API standards as an agreed specification of rules and guidelines about how to implement software interfaces and data encodings13. Geospatial software vendors, developers and users collaborate in the OGC's consensus process to develop and agree on standards that enable information systems to exchange geospatial information and instructions for geoprocessing. OGC standards are open standards. The OGC interface standards are also available in the REST style, and cover a number of aspects: • Visualisation standards e.g. Web Map Service (WMS). • Data Access Standards e.g. Web Feature Service (WFS), SensorThings API. •

Processing Standards e.g. Web Processing Service (WPS). • Metadata and Catalogue Service Standards e.g. Catalog Service for the Web (CWS). • The informatics contract between the client code which manipulates normalized data structures of geographic information based on the published API and the library code, e.g. the GeoAPI Implementation Standard. The standards above are part of the few globally agreed specifications adopted by the Technical Committee 211 of the International Organization for Standardization (ISO). ISO is also known to be working on standards in other sectors, notably in Financial Services with ISO 2002214: however, because of they are work in progress, details about them are still limited. Whilst standards of this formal and specific nature are used in the EU, there is clear evidence that the need for harmonising APIs lifecycle has been recognised. For example, the UK Government Digital Service recognised that departments were developing APIs using different tools, platforms and approaches15, and have set about working with industry to create a set of common principles for API design. The output has been a set of guidelines on how developers working in any UK public sector organisation should build APIs16 to ensure consistency, and success. Although they are titled as a ‘standard’, they are generic, and not exact or specific in the way that an ISO or OGC standard is. Nevertheless, given the fact that government is increasingly using APIs to automate processes and provide citizens with access to new services it is hoped this approach will make integration simpler and faster.

Chapter 2

APIs in the public Sector

2.1 Reasonably long section title

I’m going to randomly include a picture Figure 2.1.

If you have trouble viewing this document contact Krishna at: kks32@cam.ac.uk or raise an issue at https://github.com/kks32/phd-thesis-template/

Enumeration

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed vitae laoreet lectus. Donec lacus quam, malesuada ut erat vel, consectetur eleifend tellus. Aliquam non feugiat lacus. Interdum et malesuada fames ac ante ipsum primis in faucibus. Quisque a dolor sit amet dui malesuada malesuada id ac metus. Phasellus posuere egestas mauris, sed porta arcu vulputate ut. Donec arcu erat, ultrices et nisl ut, ultricies facilisis urna. Quisque iaculis, lorem non maximus pretium, dui eros auctor quam, sed sodales libero felis vel orci. Aliquam neque nunc, elementum id accumsan eu, varius eu enim. Aliquam blandit ante et ligula tempor pharetra. Donec molestie porttitor commodo. Integer rutrum turpis ac erat tristique cursus. Sed venenatis urna vel tempus venenatis. Nam eu rhoncus eros, et condimentum elit. Quisque risus turpis, aliquam eget euismod id, gravida in odio. Nunc elementum nibh risus, ut faucibus mauris molestie eu. Vivamus quis nunc nec nisl vulputate fringilla. Duis tempus libero ac justo laoreet tincidunt. Fusce sagittis gravida magna, pharetra venenatis mauris semper at. Nullam eleifend felis a elementum sagittis. In vel turpis eu metus euismod tempus eget sit amet tortor. Donec eu rhoncus libero, quis iaculis lectus. Aliquam erat volutpat. Proin id ullamcorper tortor. Fusce vestibulum a enim non volutpat. Nam ut interdum nulla. Proin lacinia felis malesuada arcu aliquet fringilla. Aliquam condimentum, tellus eget maximus

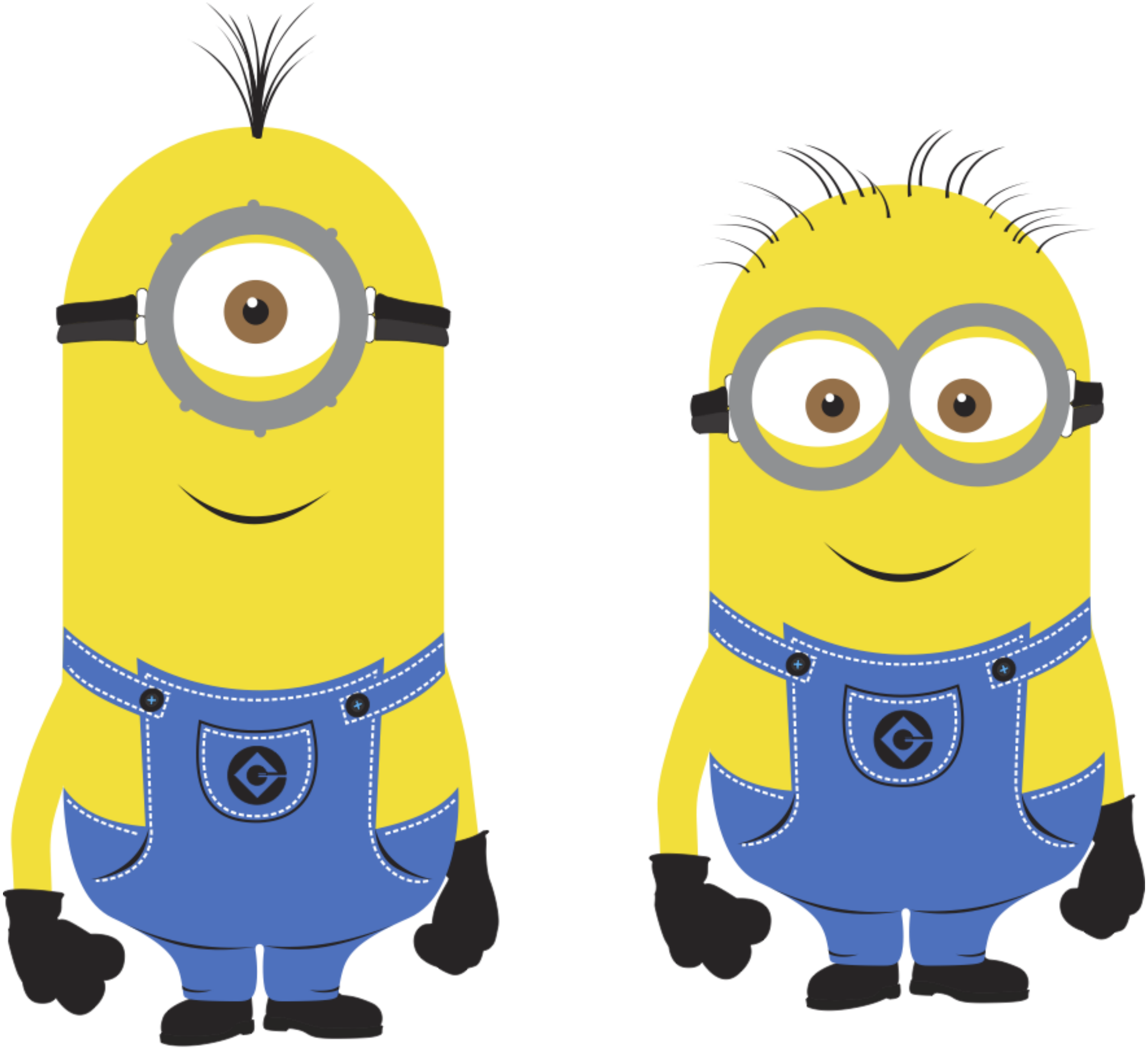


Fig. 2.1 This is just a long figure caption for the minion in Despicable Me from Pixar

porttitor, quam sem luctus massa, eu fermentum arcu diam ac massa. Praesent ut quam id leo molestie rhoncus. Praesent nec odio eget turpis bibendum eleifend non sit amet mi. Curabitur placerat finibus velit, eu ultricies risus imperdiet ut. Suspendisse lorem orci, luctus porta eros a, commodo maximus nisi.

Nunc et dolor diam. Phasellus eu justo vitae diam vehicula tristique. Vestibulum vulputate cursus turpis nec commodo. Etiam elementum sit amet erat et pellentesque. In eu augue sed tortor mollis tincidunt. Mauris eros dui, sagittis vestibulum vestibulum vitae, molestie a velit. Donec non felis ut velit aliquam convallis sit amet sit amet velit. Aliquam vulputate, elit in lacinia lacinia, odio lacus consectetur quam, sit amet facilisis mi justo id magna. Curabitur aliquet pulvinar eros. Cras metus enim, tristique ut magna a, interdum egestas nibh. Aenean lorem odio, varius a sollicitudin non, cursus a odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae;

1. The first topic is dull
2. The second topic is duller
    (a) The first subtopic is silly
    (b) The second subtopic is stupid
3. The third topic is the dullest

Morbi bibendum est aliquam, hendrerit dolor ac, pretium sem. Nunc molestie, dui in euismod finibus, nunc enim viverra enim, eu mattis mi metus id libero. Cras sed accumsan justo, ut volutpat ipsum. Nam faucibus auctor molestie. Morbi sit amet eros a justo pretium aliquet. Maecenas tempor risus sit amet tincidunt tincidunt. Curabitur dapibus gravida gravida. Vivamus porta ullamcorper nisi eu molestie. Ut pretium nisl eu facilisis tempor. Nulla rutrum tincidunt justo, id placerat lacus laoreet et. Sed cursus lobortis vehicula. Donec sed tortor et est cursus pellentesque sit amet sed velit. Proin efficitur posuere felis, porta auctor nunc. Etiam non porta risus. Pellentesque lacinia eros at ante iaculis, sed aliquet ipsum volutpat. Suspendisse potenti.

Ut ultrices lectus sed sagittis varius. Nulla facilisi. Nullam tortor sem, placerat nec condimentum eu, tristique eget ex. Nullam pretium tellus ut nibh accumsan elementum. Aliquam posuere gravida tellus, id imperdiet nulla rutrum imperdiet. Nulla pretium ullamcorper quam, non iaculis orci consectetur eget. Curabitur non laoreet nisl. Maecenas lacinia, lorem vel tincidunt cursus, odio lorem aliquet est, gravida auctor arcu urna id enim. Morbi accumsan bibendum ipsum, ut maximus dui placerat vitae. Nullam pretium ac tortor nec venenatis. Nunc non aliquet neque.

Itemize

- The first topic is dull
- The second topic is duller
  - The first subtopic is silly
  - The second subtopic is stupid
- The third topic is the dullest

Description

The first topic is dull

The second topic is duller

> **The first subtopic** is silly
>
> **The second subtopic** is stupid

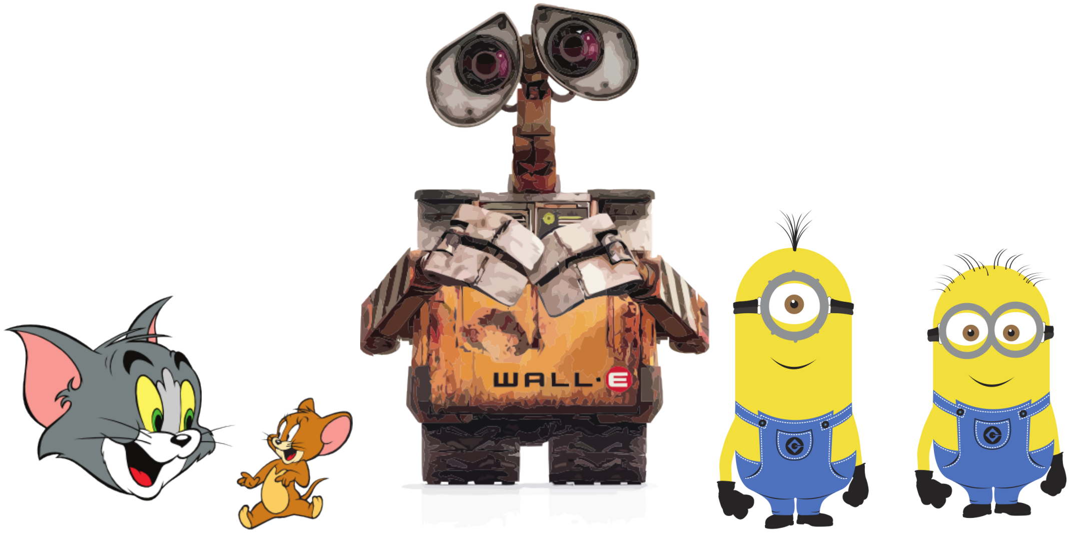
The third topic is the dullest

2.2 Hidden section

Lorem ipsum dolor sit amet, *consectetur adipiscing elit.* In magna nisi, aliquam id blandit id, congue ac est. Fusce porta consequat leo. Proin feugiat at felis vel consectetur. Ut tempus ipsum sit amet congue posuere. Nulla varius rutrum quam. Donec sed purus luctus, faucibus velit id, ultrices sapien. Cras diam purus, tincidunt eget tristique ut, egestas quis nulla. Curabitur vel iaculis lectus. Nunc nulla urna, ultrices et eleifend in, accumsan ut erat. In ut ante leo. Aenean a lacinia nisl, sit amet ullamcorper dolor. Maecenas blandit, tortor ut scelerisque congue, velit diam volutpat metus, sed vestibulum eros justo ut nulla. Etiam nec ipsum non enim luctus porta in in massa. Cras arcu urna, malesuada ut tellus ut, pellentesque mollis risus.Morbi vel tortor imperdiet arcu auctor mattis sit amet eu nisi. Nulla gravida urna vel nisl egestas varius. Aliquam posuere ante quis malesuada dignissim. Mauris ultrices tristique eros, a dignissim nisl iaculis nec. Praesent dapibus tincidunt mauris nec tempor. Curabitur et consequat nisi. Quisque viverra egestas risus, ut sodales enim blandit at. Mauris quis odio nulla. Cras euismod turpis magna, in facilisis diam congue non. Mauris faucibus nisl a orci dictum, et tempus mi cursus.

Etiam elementum tristique lacus, sit amet eleifend nibh eleifend sed [1]. Maecenas dapibu augue ut urna malesuada, non tempor nibh mollis. Donec sed sem sollicitudin, convallis velit aliquam, tincidunt diam. In eu venenatis lorem. Aliquam non augue porttitor tellus faucibus porta et nec ante. Proin sodales, libero vitae commodo sodales, dolor nisi cursus magna, non tincidunt ipsum nibh eget purus. Nam rutrum tincidunt arcu, tincidunt vulputate mi sagittis id. Proin et nisi nec orci tincidunt auctor et porta elit. Praesent eu dolor ac magna cursus euismod. Integer non dictum nunc.

[1] My footnote goes blah blah blah! ...



(a) Tom and Jerry

(b) Wall-E

(c) Minions

Fig. 2.2 Best Animations

Subplots

I can cite Wall-E (see Fig. 2.2b) and Minions in despicable me (Fig. 2.2c) or I can cite the whole figure as Fig. 2.2

Chapter 3

My third chapter

3.1 First section of the third chapter

And now I begin my third chapter here ...

And now to cite some more people Read [6], Ancey et al. [2]

3.1.1 First subsection in the first section

... and some more

3.1.2 Second subsection in the first section

... and some more ...

First subsub section in the second subsection

... and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it ...

3.1.3 Third subsection in the first section

... and some more ...

First subsub section in the third subsection

... and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it and some more and some more and some more and some more and some more and some more and some more ...

Second subsub section in the third subsection

. . . and some more in the first subsub section otherwise it all looks the same doesn't it? well we can add some text to it . . .

3.2 Second section of the third chapter

and here I write more . . .

3.3 The layout of formal tables

This section has been modified from "Publication quality tables in LaTeX*" by Simon Fear.

The layout of a table has been established over centuries of experience and should only be altered in extraordinary circumstances.

When formatting a table, remember two simple guidelines at all times:

1. Never, ever use vertical rules (lines).

2. Never use double rules.

These guidelines may seem extreme but I have never found a good argument in favour of breaking them. For example, if you feel that the information in the left half of a table is so different from that on the right that it needs to be separated by a vertical line, then you should use two tables instead. Not everyone follows the second guideline:

There are three further guidelines worth mentioning here as they are generally not known outside the circle of professional typesetters and subeditors:

3. Put the units in the column heading (not in the body of the table).

4. Always precede a decimal point by a digit; thus 0.1 *not* just .1.

5. Do not use 'ditto' signs or any other such convention to repeat a previous value. In many circumstances a blank will serve just as well. If it won't, then repeat the value.

A frequently seen mistake is to use '\begin{center}' . . . '\end{center}' inside a figure or table environment. This center environment can cause additional vertical space. If you want to avoid that just use '\centering'

Table 3.1 A badly formatted table

	Species I		Species II	
Dental measurement	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

Table 3.2 A nice looking table

Dental measurement	Species I		Species II	
	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

Table 3.3 Even better looking table using booktabs

Dental measurement	Species I		Species II	
	mean	SD	mean	SD
I1MD	6.23	0.91	5.2	0.7
I1LL	7.48	0.56	8.7	0.71
I2MD	3.99	0.63	4.22	0.54
I2LL	6.81	0.02	6.66	0.01
CMD	13.47	0.09	10.55	0.05
CBL	11.88	0.05	13.11	0.04

References

[1] Abramovich, Y. A., Aliprantis, C. D., and Burkinshaw, O. (1995). Another characterization of the invariant subspace problem. *Operator Theory in Function Spaces and Banach Lattices*. The A.C. Zaanen Anniversary Volume, *Operator Theory: Advances and Applications*, 75:15–31. Birkhäuser Verlag.

[2] Ancey, C., Coussot, P., and Evesque, P. (1996). Examination of the possibility of a fluid-mechanics treatment of dense granular flows. *Mechanics of Cohesive-frictional Materials*, 1(4):385–403.

[3] Aupetit, B. (1991). *A Primer on Spectral Theory*. Springer-Verlag, New York.

[4] Conway, J. B. (1990). *A Course in Functional Analysis*. Springer-Verlag, New York, second edition.

[5] Ljubič, J. I. and Macaev, V. I. (1965). On operators with a separable spectrum. *Amer. Math. Soc. Transl. (2)*, 47:89–129.

[6] Read, C. J. (1985). A solution to the invariant subspace problem on the space l_1. *Bull. London Math. Soc.*, 17:305–317.

Appendix A

How to install LaTeX

Windows OS

TeXLive package - full version

1. Download the TeXLive ISO (2.2GB) from https://www.tug.org/texlive/
2. Download WinCDEmu (if you don't have a virtual drive) from http://wincdemu.sysprogs.org/download/
3. To install Windows CD Emulator follow the instructions at http://wincdemu.sysprogs.org/tutorials/install/
4. Right click the iso and mount it using the WinCDEmu as shown in http://wincdemu.sysprogs.org/tutorials/mount/
5. Open your virtual drive and run setup.pl

or

Basic MikTeX - TeX distribution

1. Download Basic-MiKTeX(32bit or 64bit) from http://miktex.org/download
2. Run the installer
3. To add a new package go to Start » All Programs » MikTex » Maintenance (Admin) and choose Package Manager

4. Select or search for packages to install

TexStudio - TeX editor

1. Download TexStudio from http://texstudio.sourceforge.net/#downloads
2. Run the installer

Mac OS X

MacTeX - TeX distribution

1. Download the file from https://www.tug.org/mactex/
2. Extract and double click to run the installer. It does the entire configuration, sit back and relax.

TexStudio - TeX editor

1. Download TexStudio from http://texstudio.sourceforge.net/#downloads
2. Extract and Start

Unix/Linux

TeXLive - TeX distribution

Getting the distribution:

1. TexLive can be downloaded from http://www.tug.org/texlive/acquire-netinstall.html.
2. TexLive is provided by most operating system you can use (rpm,apt-get or yum) to get TexLive distributions

Installation

1. Mount the ISO file in the mnt directory

   ```
   mount -t iso9660 -o ro,loop,noauto /your/texlive####.iso /mnt
   ```

2. Install wget on your OS (use rpm, apt-get or yum install)

3. Run the installer script install-tl.

   ```
   cd /your/download/directory
   ./install-tl
   ```

4. Enter command ‘i’ for installation

5. Post-Installation configuration:
   http://www.tug.org/texlive/doc/texlive-en/texlive-en.html#x1-320003.4.1

6. Set the path for the directory of TexLive binaries in your .bashrc file

For 32bit OS

For Bourne-compatible shells such as bash, and using Intel x86 GNU/Linux and a default directory setup as an example, the file to edit might be

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/i386-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

For 64bit OS

```
edit $~/.bashrc file and add following lines
PATH=/usr/local/texlive/2011/bin/x86_64-linux:$PATH;
export PATH
MANPATH=/usr/local/texlive/2011/texmf/doc/man:$MANPATH;
export MANPATH
```

```
INFOPATH=/usr/local/texlive/2011/texmf/doc/info:$INFOPATH;
export INFOPATH
```

Fedora/RedHat/CentOS:

```
sudo yum install texlive
sudo yum install psutils
```

SUSE:

```
sudo zypper install texlive
```

Debian/Ubuntu:

```
sudo apt-get install texlive texlive-latex-extra
sudo apt-get install psutils
```

Appendix B

Installing the CUED class file

LATEX.cls files can be accessed system-wide when they are placed in the <texmf>/tex/latex directory, where <texmf> is the root directory of the user's TEXinstallation. On systems that have a local texmf tree (<texmflocal>), which may be named "texmf-local" or "localtexmf", it may be advisable to install packages in <texmflocal>, rather than <texmf> as the contents of the former, unlike that of the latter, are preserved after the LATEXsystem is reinstalled and/or upgraded.

It is recommended that the user create a subdirectory <texmf>/tex/latex/CUED for all CUED related LATEXclass and package files. On some LATEXsystems, the directory look-up tables will need to be refreshed after making additions or deletions to the system files. For TEXLive systems this is accomplished via executing "texhash" as root. MIKTEXusers can run "initexmf -u" to accomplish the same thing.

Users not willing or able to install the files system-wide can install them in their personal directories, but will then have to provide the path (full or relative) in addition to the filename when referring to them in LATEX.