CS3012: Software Engineering Measurement Report

Dearbhla Boylan
Student ID: 17330661
E-mail: boylande@tcd.ie

Introduction

“Data analytics is the science of analysing raw data in order to make conclusions about that information.” [1] This report will focus on data analytics regarding the software engineering process.

This report has four sections: Software Metrics, Platforms Available to this Network, Algorithmic Approaches and the Ethics surrounding Analytics. Software Metrics discusses the background of these metrics and how the different metrics involved in analysing a software project can be classified. The computational platforms section discusses some of the older platforms for measuring software data as well as some of the most commonly used ones today. There are many other ways of analysing software data without the use of a platform and this is discussed in the third section of this report, algorithmic approaches this includes older and more modern methods of analysing a software team’s project. The final section of this report, ethics surrounding analytics discusses the ethics around data analytics as well as my own opinion on the topic.

[1] https://www.investopedia.com/terms/d/data-analytics.asp

1. Measurable Data

This section of the report explains the definition of a software metric, a description of the development of the use of software metrics and the different types of software metrics.

1.1 Background of Software Metrics

“A software metric is a measure of software characteristics which are quantifiable or countable.” [2]

These metrics can be used for measuring software performance, planning and measuring productivity.
The area of software metrics is quite a ‘young’ area compared to other areas in software engineering such as testing etc. in terms of research, documentation, execution and standards. A paper written in 1999 [3] details the history and development of the use of metrics in software engineering. According to this paper the main metric in software engineering projects was LOC (Lines of Code). This metric is still very much used today and is useful however there are drawbacks to it such as it doesn’t recognize complexity, functionality and other metrics like programmer productivity. This changed in the mid-1970s when a lot more research and development was completed in this area.

Software metrics are mainly used by product managers as opposed to the actual software engineers. Some of the goals of using software metrics are:

- Increase return on investment (ROI)
- Identify areas of improvement
- Manage workloads
- Reduce overtime
- Reduce costs

[2] https://stackify.com/track-software-metrics/
[3] Fenton, N. E., and Martin, N. (1999) "Software metrics: successes, failures and new directions." Journal of Systems and Software 47.2 pp. 149-157.

1.2 How to Classify Metrics

Software metrics can be divided into direct and indirect metrics. [4]

- A direct metric is a metric that does not depend on the measure of another attribute e.g. Lines of code, execution speed.
- An indirect metric relies on more than one measure e.g. programmer productivity, functionality, complexity, reliability.

Another popular way of classifying metrics is placing a metric into one of three categories: Process, Project, Product.

According to the book: Metrics and Models in Software Quality Engineering, software metrics can be organised into three categories: product metrics, process metrics and project metrics.[5]
Product metrics describe the actual product that is being developed e.g. size of product, complexity, design level. According to the online resource ecomputernotes.com [6] product metrics are used to detect and correct potential problems before they result in problems. Product metrics are also used to understand the efficiency of analysis, design, and code model, potency of test cases, overall quality of the software under development.
Process metrics can be used to improve software development and maintenance e.g. defect arrival pattern during testing. Process metrics are used to assess the efficiency of the software process that is performed using the process as a framework. The skill and motivation of the people, the complexity of the product and the level of technology used in the software development have an important influence on the quality and team performance.
Project metrics describe the project and execution e.g. number of developers, cost of project etc. Project metrics allow the project managers to assess current projects, track potential risks, identify problem areas, adjust workflow, and evaluate the project team's ability to control the quality of work products. Project metrics are used to detect any errors during development. For example, as software evolves from design

---

[4] https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know
[5] http://www.informit.com/articles/article.aspx?p=30306&seqNum=5
[6] http://ecomputernotes.com/software-engineering/classification-of-software-metrics

to coding, project metrics are collected to assess the quality of the design and obtain indicators that in turn affect the approach chosen for coding and testing. Also, project metrics are used to measure production rate, which is measured in terms of models developed, function points, and delivered lines of code.

1.3 Examples of Metrics

Some of the most commonly used software metrics today are:
Agile process metrics. Some of the most useful of these metrics are [7]:

- Code Churn
    - Code Churn is the percentage of a developer's own code representing an edit to their own recent work. It's often measured as lines of code (LOC) that were modified, added and deleted over a short period of time such as a few weeks. Code churn can indicate many things about the progress of a project in development e.g. an increase in code churn may indicate a developer is struggling to solve a problem.
- Lead Time
    - Lead Time is the time period between the beginning of a project's development and its delivery to the customer. A team's lead time history will help managers predict when an item will be ready for the customer. Lowering lead time is a way to improve how responsive software developers are to customers
- Impact
    - Impact measures the effect of a developer's code change on the software development project. The impact of a code change depends on things like how many lines are changed, how many files were changed and how complex was the code etc.

[7] https://blog.gitprime.com/5-developer-metrics-every-software-manager-should-care-about/

2. Computational Platforms Available to this Network

This section of the report provides examples of tools and platforms that product managers and engineers use to collect and analyse their data.

2.1 Leap

Leap was an automated tool that was developed after the use of the Personal Software Process (discussed in the algorithms section of this report).[8] The user can record effort, size and defect information. This tool was an improvement from the Personal Software Process however the main problem with it was that if the metrics being measured were changed then the software of leap had to be changed. Users were able to use historical data in Leap to improve planning and quality assurance.

2.2 Hack Stat

The paper, “Beyond the Personal Software Process: Metrics Collection and Analysis for the Differently Disciplined,” details the development of Hackystat which was developed after the tool Leap. According to online resources [9] “Hackystat is an open source framework for collection, analysis, visualization, interpretation, annotation, and dissemination of software development process and product data.” Hackystat uses sensors that are attached to development tools and theses sensors collect metrics about the development activity e.g. effort, size, defect etc. This data is then sent to the Hackystat SensorBase web service for storage. This web service can be queried for many reasons such as integration with communications and generate visualizations of the data. Visualization of data can be a very useful tool for identifying areas of a project that need improvement.

2.3 Git Prime

[8] P.M. Johnson et al., “Beyond the Personal Software Process: Metrics Collection and Analysis for the Differently Disciplined,” Proc. 25th Int’l Conf. Software Eng. (ICSE 03), IEEE CS, 2003, pp. 641–646

[9] http://csdl.ics.hawaii.edu/Research/Hackystat/

According to the official GitPrime website [10] “GitPrime aggregates historical git data into easy to understand insights and reports, to help make engineering teams more successful. Debug your development processes with objective data. Identify bottlenecks, compare trends, and keep a pulse on the health of your software teams.”

GitPrime is basically an organizational too that allows users to use various ways of measuring and communicating about productivity in their software engineering. It is designed for the leaders of software projects e.g. product managers, tech leads etc. It is a tool mainly used by teams as working with a lot of people can lead to difficulties. It is a secure tool and does not keep copies of the repositories using it but connects using SSH key pairs. [11] Common problems that can arise is trouble tracking the workflow of people working on the same project, this is especially the case for projects with a very large team working on it simultaneously. To combat this GitPrime has a feature called Work Log, this feature allows team leaders to view a team’s contributions. The leader can view a project’s code commits, ticket activity, and PRs on one dashboard.

[10] https://www.gitprime.com/
[11] https://help.gitprime.com/general/what-is-gitprime-exactly

3. Algorithmic Approaches

This section of the report details and explains some of the most commonly used algorithmic processes and approaches to calculating the metrics that managers and team leads are collecting and analysing.

3.1 Personal Software Process

The personal software process is one of the earliest ways of measuring the software engineering process. It was based on the principal that "to produce quality software systems, every engineer who works on the system must do quality work." [12] The goals of this process when it was first set up were to improve estimation and quality assurance. This is done by collecting size, time, and defect data on an initial set of software projects and performing various analyses. The PSP shows engineers how to plan and track their work, establish, measurable goals, track performance against these goals etc. This allows engineers to manage the quality of their project from the beginning of the development phase, they can analyse the results of each project and they can use these results when they are looking to improve.

[13]To carry out this process users print out forms where they record effort, size, and defect information. This process can cause problems because of the meticulous nature of this process e.g. every single compiler error must be recorded, this therefore can cause backlog in the analysing the part of this process.
The data collected in the PSP consists of time measures, size measures and quality measures. Time measures take a log on the time spent during each phase of the process, considering interruptions e.g. phone call. Size measures is a record of the original estimated size of the product before development begins and the final size of the product when development has finished. This measure must correlate with the time spent on development. With quality measures engineers record details on every defect they encounter. The main quality measures used in the PSP are: Defect density, Review rate, Development time ratios, Defect ratios, Yield, Defects per hour, Defect removal leverage, Appraisal to failure ratio (A/FR).

---

[12] https://resources.sei.cmu.edu/asset_files/TechnicalReport/2000_005_001_13751.pdf.

[13] P.M. Johnson et al., "Beyond the Personal Software Process: Metrics Collection and Analysis for the Differently Disciplined," Proc. 25th Int'l Conf. Software Eng. (ICSE 03), IEEE CS, 2003, pp. 641–646

3.2 Algorithmic Cost Modelling

According to the research paper, Software Cost Estimation using Algorithmic Model and Non-Algorithmic Model a Review "In Software Project Management System, the most critical activity is estimating the software cost and effort." [14]

A commonly used algorithmic software cost model is the Constructive Cost Model (COCOMO), which was developed by Barry Boehm in 1970.
[15]Cocomo (Constructive Cost Model) is based on LOC (number of Lines of Code). It is a cost estimate model for software projects. Projects are put into one of three categories : Organic, Semi-detached and Embedded. Organic is for projects where team size is small, and the problem is understood. Embedded is for projects that require highest level of complexity and experience and usually have larger teams. Semi-detached is for projects where aspects like team size and experience are on a level in-between organic and embedded. The three types of COCOMO model are: Basic , Intermediate and Detailed
The basic model is used for quick calculations but is not completely accurate because it doesn't take account of all factors.

$\mathrm{E}= \mathrm{a}(\mathrm{KLOC})^{b}$ Constants a and b vary depending on the type of system. KLOC represents lines of code.

The intermediate considers of cost drivers : Product attributes, hardware attributes, Personnel attributes and Project attributes. The leader of the project rates these factors on a scale of 1-3 and then based on the ratings takes values from the table below. These values are then multiplied together to make the EAF (Effort Adjustment Factor.

$$\mathrm{E}= (\mathrm{a}(\mathrm{KLOC})^{b}) * EAF$$

In detailed COCOMO, the whole software is divided into different modules and then COCOMO is applied in different modules to estimate effort and then sum the effort.

The Six phases of detailed COCOMO are:

1. Planning and requirements
2. System design
3. Detailed design
4. Module code and test
5. Integration and test
6. Cost Constructive model

The effort is calculated as a function of program size and a set of cost drivers are given according to each phase of the software lifecycle.

[14] https://research.ijcaonline.org/itcce/number2/ITCCE2010.pdf
[15] https://www.geeksforgeeks.org/software-engineering-cocomo-model/

Table of Cost Drivers for COCOMO [16]

	RATING				
Cost Drivers	**Very Low**	**Low**	**Normal**	**High**	**Very High**
Product Attributes					
RELY, required reliability	.75	.88	1.00	1.15	1.40
DATA, database size	.94	1.00	1.08	1.16	
CPLX, product complexity	.70	.85	1.00	1.15	1.30
Computer Attributes					
TIME, execution time constraint			1.00	1.11	1.30
STOR, main storage constraint			1.00	1.06	1.21
VITR, virtual machine volatility		.87	1.00	1.15	1.30
TURN, computer turnaround time		.87	1.00	1.07	1.15
Personnel Attributes					
ACAP, analyst capability	1.46	1.19	1.00	.86	.71
AEXP, application experience	1.29	1.13	1.00	.91	.82
PCAP, programmer capability	1.42	1.17	1.00	.86	.70
VEXP, virtual machine experience	1.21	1.10	1.00	.90	
LEXP, Programming language experience	1.14	1.07	1.00	.95	
Project Attributes					
MODP, modern programming practices	1.24	1.10	1.00	.91	.82
TOOL, use of SW tools	1.24	1.10	1.00	.91	.83
SCHED, development schedule	1.23	1.08	1.00	1.04	1.10

Table of Constants for a and b

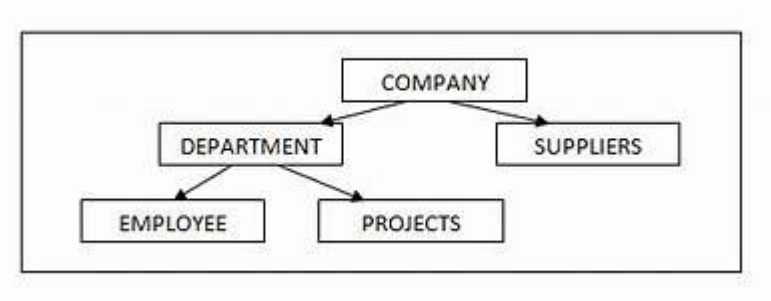
Projects	Basic a	Basic b	Inter a	Inter b
Organic	2.4	1.05	3.2	1.05
Semi-detached	3.0	1.12	3.0	1.12
Embedded	3.6	1.20	2.8	1.20

16 http://education.dewsoftoverseas.com/QE/QuickReference/Software%20Enginering/images/3.3.as5.gif

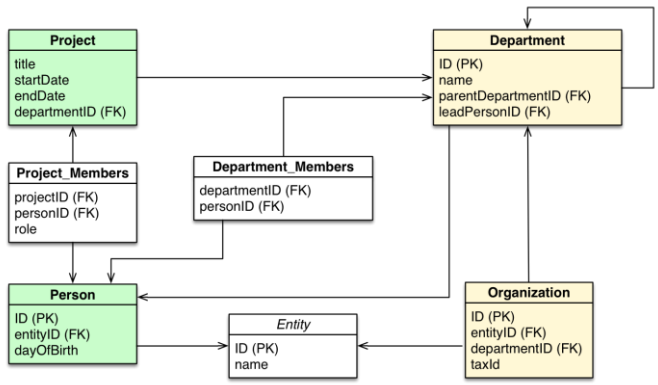
3.3 Data Processing Models

Data processing models or data modelling has become a very popular method of analysing projects before any actual coding begins. According to online resources data modelling is "the process of documenting a complex software system design as an easily understood diagram, using text and symbols to represent the way data needs to flow". [17] A data model basically shows the relationships among data. Examples of some data models include: Hierarchical data modelling, relational data modelling, entity relationship model, graph data models.

- Hierarchical Modelling [18]



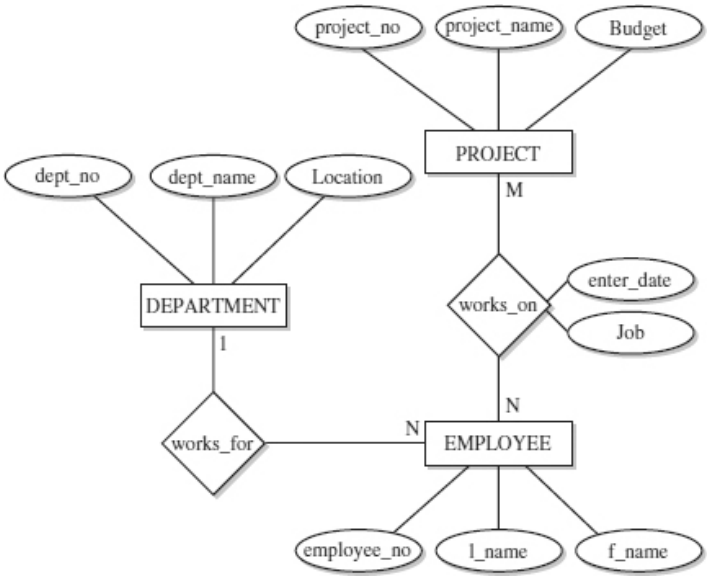
- Relational Data Modelling [19]



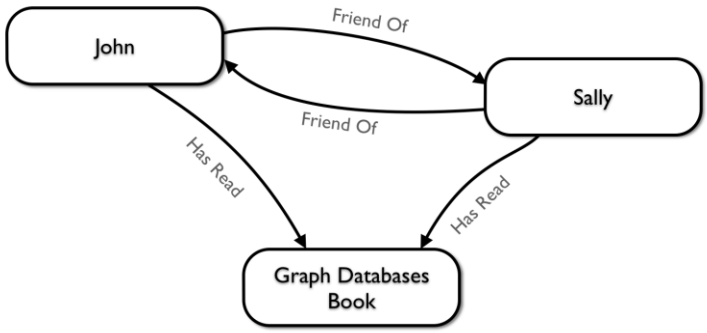
- Entity Relationship Model[20]

[17] https://searchdatamanagement.techtarget.com/definition/data-modeling
[18] https://www.tutorialcup.com/images/dbms/record-based-data-models/Hierarchical-Data-Models.png

[19] https://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/organization_relational.png
[20] http://logicalread.com/media/427690/0052_001.jpg



- Graph Data Model [21]



The diagrams above help show engineers how different components of a system interact with each other. This helps to reduce problems encountered during development as potential problems can be spotted beforehand. For example, the diagram under relational data modelling is a class diagram which is used to show the intended structure of a system. The entity relationship diagram shows the relationships between different entities in a database. [22]

[21] http://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/data-modeling-1.png
[22] https://www.smartdraw.com/entity-relationship-diagram/

3.4 Computational Intelligence

"Computational Intelligence (CI) is the theory, design, application and development of biologically and linguistically motivated computational paradigms." [23] Fuzzy sets, rough sets and interval analysis are all commonly used aspects of computational intelligence.

Computational intelligence has become a method used in assessing the quality of a software project.
According to the paper, "The Role of Computational Intelligence in Quantitative Software Engineering", computational intelligence can be used to help improve software quality, structure and interpretation of results. CI can recognise the variety of software systems, quality, indexes and processes. CI can help accommodate non-numeric data which is more useful in assessing the quality of results and creating feedback. [24]

A lot of research has been completed in the area of computational intelligence in the last number of years including the area of software engineering. Recent studies have found that CI can help managers decide whether software is ready to be released or not. [25] This is because CI can be used as a method of predicting faults and faults in testing. This paper states that there are CI models that can predict:

- the number of faults (defects),
- the amount of code changes required to correct a fault and
- the amount of time (in minutes) to make the changes in respective object classes using software metrics

This can help improve the estimation and predictiveness of whether software is ready or not for release by determining if the quality of the software is of a high enough standard.

[23] https://cis.ieee.org/about/what-is-ci
[24] https://www.researchgate.net/publication/292463091_The_Role_of_Computational_Intelligence_in_Quantitative_Software_Engineering
[25] http://www.engineeringletters.com/issues_v14/issue_2/EL_14_2_11.pdf?origin=publication_detail

4. Ethics Surrounding Analytics

The ethics behind the collection, protection and use of data has become a popular topic of discussion. Now that people are becoming more aware of how much data is being collected the organizations behind it are being criticised and their methods are being called into question. I guess the main questions now circulating are: how much data should be collected ? What kind of data should be collected ? Who should be able to access the data? What exactly is considered a breach of individual privacy and security?

4.1 Background to Data Protection: GDPR

In April 2016 the European Union approved the General Data Protection Regulation and in May 2019 this regulation was enforced. These regulations were brought into place to enforce data privacy laws across Europe, to protect the data privacy of all EU citizens and to encourage all organizations in the region of the EU to change their approach to data privacy. [26]

However, there is still a lot to learn about in the area of data privacy, especially regarding determining the line between what data can be collected and what is considered a breach of security and privacy.

There have been many scandals in recent years that have caused many people to have serious concerns about how much of their data is being used and what it is being used for. The actions of Cambridge Analytica have caused a huge discussion worldwide in the area of data privacy and data security and have put companies like Facebook under scrutiny.

4.2 Monitoring

Traditionally when measuring data for software projects companies mainly focused on the code being written but now not only are tech companies monitoring their engineer's code and the quality of their code etc. but they are now monitoring the actual employees themselves.

[26] https://eugdpr.org/

In 2015 30% of large employers were monitoring their employees in non-traditional ways e.g. analysing email, tracking movement. By 2018 this number had increased from 30% to 46% and it was projected to reach over 50% in 2019. This article states that the numbers of employees who accept monitoring is increasing mainly when it's related to their job and performance goals. However, in a poll carried out by the HR Metrics and Analytics Summit, there was a strong majority of people (72%) against the monitoring of their private social media accounts. [27] I think it is fair and that people have the right to keep their personal accounts private however I can understand how analysing work emails and movement in the workplace could lead to a better understanding of where productivity could be improved, and difficulties employees are having. I think employees must make sure they were only using their work/company email for work related communications and not using their company email for any other topic. As work emails could also be screened to make sure they don't contain inappropriate material.

I don't think many people realise how much of what they do can be measured., For example keystrokes can be measured, there are programs that can alert a supervisor if a keyboard has been idle for a prolonged period, websites can be flagged and blocked. [28] The list is extensive.

There are pros and cons to employee monitoring. As seen in the earlier sections of this report measuring certain types of data can prove to be useful and make the development process of projects more efficient.
Although most companies have good intentions when it comes to monitoring their employees and they do it with the intention of improving productivity and helping their business however there are drawbacks to it is well: employees can feel too controlled and it undermines the trust between the employees and the employers. [29]
It was for these reasons that the company Amazon have been under scrutiny in recent years. In 2016 Amazon applied for a patent for a wristband to monitor their employees and in 2018 they were awarded this patent. [30] The wristband includes a feature which would "buzz" if for example a worker went to pick up the incorrect item. When this news was revealed there were concerns that this was another level of surveillance and it could make employees feel like robots rather than people. [31]

This kind of surveillance could negatively affect an employee's mental health. There has been research completed that suggests when employees are under surveillance,

[27] https://www.shrm.org/hr-today/news/all-things-work/Pages/watching-the-workers.aspx
[28] https://www.shrm.org/hr-today/news/hr-magazine/Pages/0615-employee-monitoring.aspx
[29] https://recruitingblogs.com/profiles/blogs/how-do-big-companies-monitor-their-employees
[30] https://www.theverge.com/2018/2/1/16958918/amazon-patents-trackable-wristband-warehouse-employees
[31] https://www.nytimes.com/2018/02/01/technology/amazon-wristband-tracking-privacy.html

it can cause heightened stress levels in the workplace : “For 23 percent of UK employees, these IT systems are used to check the quality of work produced. Feelings of exhaustion and anxiety related to work are 7.5 percent higher among these 23 percent”. [32] This is concerning to me as employers are obligated to provide a safe and secure working environment however using these surveillance methods compromises this.

4.3 Conclusion: My Opinion

I do agree with the statement that monitoring employees has benefits and it does have to be done to some extent. But once the monitoring begins to make employees feel undermined or negatively affect their attitude towards work then I think it has gone too far. I think employers have the right to monitor employee conversations that are being held through company provided emails or another communication tool, but employers should not be monitoring any personal devices, social media accounts or emails. Ultimately, we all have the right to privacy and when a company is measuring data that an employee is uncomfortable with then they should be able to say no.

[32] https://www.zdnet.com/article/workplace-surveillance-boosts-stress-levels/

Bibliography

- https://www.investopedia.com/terms/d/data-analytics.asp
- https://stackify.com/track-software-metrics/
- Fenton, N. E., and Martin, N. (1999) "Software metrics: successes, failures and new directions." Journal of Systems and Software 47.2 pp. 149-157.
- https://maisqual.squoring.com/wiki/index.php/Software_Engineering_Metrics:_What_Do_They_Measure_And_How_Do_We_Know
- http://www.informit.com/articles/article.aspx?p=30306&seqNum=5
- http://ecomputernotes.com/software-engineering/classification-of-software-metrics
- https://blog.gitprime.com/5-developer-metrics-every-software-manager-should-care-about/
- http://csdl.ics.hawaii.edu/Research/Hackystat/
- P.M. Johnson et al., "Beyond the Personal Software Process: Metrics Collection and Analysis for the Differently Disciplined," Proc. 25th Int'l Conf. Software Eng. (ICSE 03), IEEE CS, 2003, pp. 641–646
- https://www.gitprime.com/
- https://help.gitprime.com/general/what-is-gitprime-exactly
- https://resources.sei.cmu.edu/asset_files/TechnicalReport/2000_005_001_13751.pdf.
- https://research.ijcaonline.org/itcce/number2/ITCCE2010.pdf
- https://www.geeksforgeeks.org/software-engineering-cocomo-model/
- http://education.dewsoftoverseas.com/QE/QuickReference/Software%20Engineering/images/3.3.as5.gif
- https://searchdatamanagement.techtarget.com/definition/data-modeling
- https://www.tutorialcup.com/images/dbms/record-based-data-models/Hierarchical-Data-Models.png
- https://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/organization_relational.png
- http://logicalread.com/media/427690/0052_001.jpg
- http://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/data-modeling-1.png
- https://www.smartdraw.com/entity-relationship-diagram/
- https://cis.ieee.org/about/what-is-ci
- https://www.researchgate.net/publication/292463091_The_Role_of_Computational_Intelligence_in_Quantitative_Software_Engineering
- http://www.engineeringletters.com/issues_v14/issue_2/EL_14_2_11.pdf?origin=publication_detail
- https://eugdpr.org/
- https://www.shrm.org/hr-today/news/all-things-work/Pages/watching-the-workers.aspx
- https://www.shrm.org/hr-today/news/hr-magazine/Pages/0615-employee-monitoring.aspx
- https://recruitingblogs.com/profiles/blogs/how-do-big-companies-monitor-their-employees
- https://www.theverge.com/2018/2/1/16958918/amazon-patents-trackable-wristband-warehouse-employees

- https://recruitingblogs.com/profiles/blogs/how-do-big-companies-monitor-their-employees
- https://www.zdnet.com/article/workplace-surveillance-boosts-stress-levels/