GIMA
Geographical Information Management and Applications

Final Thesis Report

Using Semantic Technologies for Effective GIS Workflow Translation

Author:
Alexandra Rowland
a.c.s.j.rowland@students.uu.nl

Supervisor:
Dr. Simon Scheider
s.scheider@uu.nl

Responsible Professor:
Prof. dr Stan Geertman
s.c.m.geertman@uu.nl

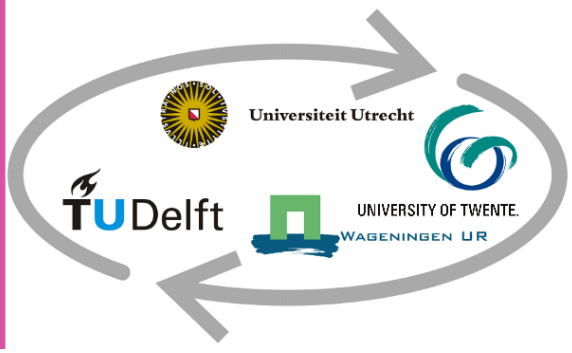


Table of Contents

List of Abbreviations

1. GIS – Geographical Information Systems or Sciences
2. SWfMS – Scientific Workflow Management System
3. CCDT – Core Concept Data Types
4. WfMS – Workflow Management System
5. SLTL – Semantic Linear Time Logic
6. PC4 – Postcode Area 4
7. XML – Extensible Markup Language
8. RDF – Resource Description Framework
9. OWL – Ontology Web Language
10. URI – Unique Resource Identified
11. URL – Unique Resource Locator
12. RDFS – Resource Description Framework Schema
13. APE – Advanced Pipeline Engine
14. SOTA – State-Of-The-Art
15. DAG – Direct Acyclic Graph
16. WF – Workflow
17. LCS – Least Common Subsumer
18. CBS – Centraal Bureau voor de Statistiek
19. dB – Decibels

List of Figures

List of Tables

List of Equations

Abstract

The use of geospatial workflow management systems to assist users with the composition of often complex GIS workflows is not without precedent. However, even where these management systems are used, users are still required to have an intricate knowledge of the geospatial domain, specific GIS tools and functions that these workflows are made up of as well as the particular software environment that the workflow is being constructed in; limiting the user group of these management systems to GIS experts and those which have become familiar with the software being used and, therefore, limiting the accessibility of the GIS domain and the meaningful answering of geospatial questions in a range of contexts. A promising solution to these limitations is the automatic synthesis of workflows making use of semantic technologies to support this process. However, the range of software environments available for carrying out geospatial analysis means that workflows created or synthesised for one environment are currently not easily translated into another environment without further knowledge of the particularities of each; presenting another challenge to the accessibility of the GIS field and the production of meaningful spatial analysis. Building on the body of work available for GIS workflow synthesis, the main research objective of this thesis is to present a novel approach to GIS workflow translation supported by the formalisation of domain knowledge using semantic technologies where GIS workflows are translated between two GIS software environments, ArcGIS and QGIS. Workflows are generated in and translated between each environment by the translation mechanism based on five predefined geo-analytical questions where expert generated examples of the workflows answering these questions are used as the basis of the quality and similarity assessments of the workflows translated as part of this research. The research presented herein highlights the promising capacity of the translation mechanism developed through this project to produce workflows with both a reasonable level of quality and to translate workflows with a reasonable level of similarity in order to answer the spatial questions posed.

Chapter 1 Introduction

Since the identification of the geographic information system (GIS) in 1968 (Tomlinson, 1969), the field of geographic information sciences and its associated technologies have grown rapidly and now include geostatistics and -coding, network analysis and various types of geographic related modelling. The technology is currently available both commercially and in open source across a range of software environments, including Esri's ArcGIS, Grass GIS, QGIS and R's geo-spatial components, and has been adopted in several contexts ranging from public management and healthcare to real estate and environmental planning. As the field has expanded, so too has the complexity, distribution, and functionality of the tools increased, making the use of a GIS and its technological outputs increasingly complex, particularly for the non-GIS expert. Indeed, although the main function of these technologies is to assist with the answering of spatial questions, the availability of geo-analytical tools across various software environments often hinders the meaningfulness of the analysis where tools and processes are not known to or fully understood by the user.

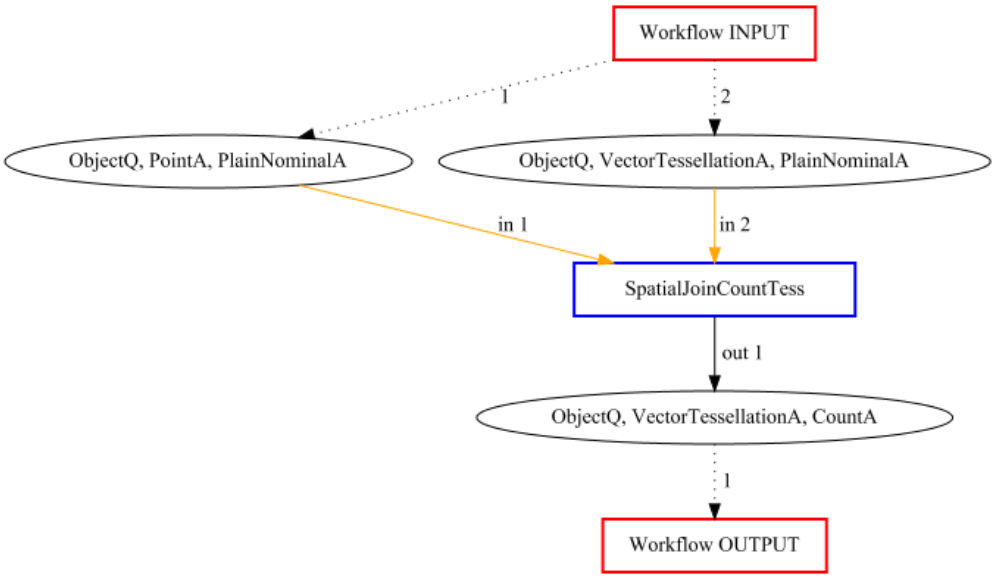
One solution to this increasing complexity, as adopted in other fields facing rapid technological expansion, is the widespread use of workflows to manage processes and the construction of these using scientific workflow management systems (SWfMS). SWfMSs improve the clarity of the available resources, simplify workflow construction when carrying out analysis and allow for storing and reuse of the workflows (Lin et al., 2008; Lamprecht, Margaria & Steffen, 2009). One of the most well-known implementations of this management system for GIS processes is Esri's ArcGIS Model Builder. The use of workflows in the scientific process allows analysts to gain a clear overview of the available resources, the connections between tools as well as allow for workflows and processes to be saved in the interest of reuse and replication (Chen et al., 2003; Lamprecht, Margaria & Steffen, 2009). A further and more recent innovation is the development of mechanisms which automate the synthesis of GIS workflows. This has been explored and implemented to different extents across the field with more recent work exploring the use of semantic technologies for the purpose of workflow synthesis (Steffen et al., 1993; Lamprecht et al., 2010; Kasalica & Lamprecht, 2018; Meerlo, 2019; Kruiger et al., 2020). The usefulness of these SWfMSs for the reduction of the complexity around the technology and improving workflow reuse, particularly those which automate or simplify the construction of workflows, has been highlighted across the relevant body of literature (Chen et al., 2003; Zhang et al., 2007; Lamprecht, Margaria & Steffen, 2009).

Although there is clear evidence that the use of workflow management systems used in the geographical sciences can improve the accessibility of GIS technologies for non-experts, this usefulness is often limited to the management software in which the workflow was originally constructed or synthesised. In generating GIS workflows, the management system naturally dictates the naming conventions of tools, data movement processes and connection conventions, data transformations, analysis and final data visualisations for all workflows created. It is these functionality silos which make the sharing and re-use of workflows across a range of available software environments difficult; particularly for less skilled, non-expert GIS users; essentially becoming a barrier to the efficiency of the scientific process and potentially the meaningfulness of GIS analysis. It is based on these barriers that the ability to translate GIS workflows between different software environments becomes necessary and is not yet explicitly discussed or pursued in the literature.

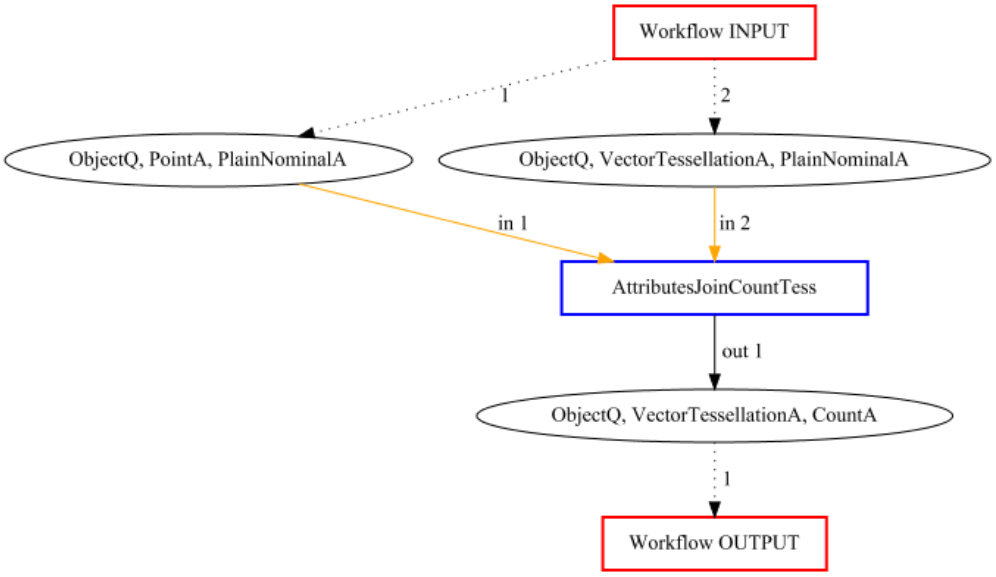
It is within this gap in the literature that this thesis situates itself. It is argued here that the use of semantic technologies to develop a translation mechanism which describes and abstracts GIS tools to their input and output types and allows for the substituting of functionally equivalent tools across software packages supports the scientific process in removing the barriers to workflow translation. Workflow translation is important because it makes the use of GIS technologies more accessible to a greater number of people; experts, and non-experts alike. It also allows for workflows to be shared, repurposed, and reapplied to a range of contexts without the need to know the functional requirements of a particular software environment. Building on the literature and innovations seen using semantic technologies for automatic workflow synthesis, workflow translation would support the use, reuse and sharing of GIS workflows across software environments and contexts where extensive knowledge of both environments is not present. Additionally, and perhaps more interestingly, the use of semantic technologies developed for the purpose of workflow translation could also support the generation of a cross-platform tool repository in the future and cover various GIS software environments, making tools easily comparable and choices around software for different functional needs easier. Indeed, a vision for such a repository would be that tools can simply be retrieved based on the required inputs and outputs of an analysis and workflows easily and quickly synthesised.

The development of this translation mechanism requires the development of technical elements and the configuration of a synthesis engine to generate workflows in these environments. The automatic synthesis mechanisms already implemented on GIS workflows (Kruiger et al., 2020) are well placed to assist with the translation of GIS workflows and will be used herein. To highlight the potential that this research has to support the realisation of GIS workflow translation, the mechanism needs to be evaluated for quality and generated workflows assessed for similarity with the overarching goal of understanding whether these are able to produce quality workflows which can be implemented with meaningful analytical results in a range of GIS software environments. As part of this research, the use of ontology-based semantic workflow composition to translate workflow between software environments will be tested, highlighting the relevance of semantic technologies for this purpose. The following figure (Figure 1.1) highlights how a workflow for a given question would look in ArcGIS and QGIS based on expert knowledge of both systems and following the abstraction of the tools into data types through the proposed translation mechanism. Although this research relies on expert knowledge for evaluation, it is hoped that the outcome will highlight the potential that translation in this way has to improve accessibility and sharing of GIS workflows. To achieve these research goals, the following steps will be taken over the course of this research project.

Figure 1.1 Example Workflows Translated Using Proposed Translation Mechanism



a. ArcGIS workflow abstracted into workflow inputs and outputs for the question "What is the number of medical facilities (pharmacies, clinics, hospitals etc.) in each PC4 area?"



b. QGIS workflow abstracted into workflow inputs and outputs for the question "What is the number of medical facilities (pharmacies, clinics, hospitals etc.) in each PC4 area?"

Firstly, the research goals, objectives and questions will be outlined in Chapter 2. These will dictate the chronologicity of this research and guide the identification and implementation of relevant technical components and the necessary evaluations and assessments for achieving the research goals. Chapter 3 will outline the existing literature on which this research will build. The theoretical framework will cover workflows and workflow management systems, focusing specifically on scientific workflows, GIS

workflow synthesis and translation as well as available methods for workflow quality evaluation, ontology evaluation and workflow similarity assessment. Here, the CCDT ontology (Scheider et al., 2020) which has been previously used in workflow synthesis will be discussed with the intention of making use of this ontology in this research. Chapter 4 will comprehensively outline the research methodology implemented as part of this research thesis. Here, in addition to other methodological steps, the competency questions which form the basis of so many these steps will be outlined in line with a fictional GIS analytical scenario, namely, an Amsterdam Liveability Atlas based on a limited set of analytical criteria. The final chapters of this document will outline the results of this research as well as the points of further research.

Chapter 2 Research Objectives

The aim of this research thesis, in highlighting the relevance of semantic web technologies for effective GIS workflow translation, is threefold and will be undertaken in several stages. The first goal of this thesis is to develop a novel approach to the translation of GIS workflows using an ontology-based translation mechanism. The workflows generated, implemented, and translated using the translation mechanism will be constructed using a defined set of geo-analytical questions, workflow goal and task specifications developed using realistic GIS project scenario designed for this purpose. The creation of this mechanism requires the identification of necessary technical elements related, including ontologies and necessary annotations, for workflow translation and the availability of these elements within the GIS field. The second goal of this thesis is the evaluation of both the ontology used in the research and the quality of the translated workflows using the translation mechanism developed. The evaluation methods used in this thesis are identified following a thorough review of relevant literature. Lastly, the third goal of this research thesis is the assessment of the similarity between those workflows manually generated and those generated as a result of the implementation of the translation mechanism in order to measure the functional differences between software environments. The similarity assessment will ideally cover both the similarity between the tools used in the workflows across different geo-analytical software environments and the similarity of the workflow outputs once implemented. The following research objectives, questions and sub-questions closely follow the research goals identified.

3.1 Research Objectives

1. Develop ontology-based methods that allow for the meaningful, automatic translation of GIS workflows between two geo-analytical software environments for a given set of geo-analytical questions.
2. Evaluate the quality of the automatically translated workflows in order to evaluate the capacity of the translation mechanism and the functional aspect of the ontology which was developed for this purpose.
3. Assess the similarity between the translated workflows, including the similarity of both tools and outputs, for those translations produced manually and those produced automatically using a given similarity assessment technique.

3.2 Research Questions

Using the above research objectives as a guide, the following research questions have been defined below. To guide the review of relevant literature and the progression through the respective stages of research, sub-questions for every research question (indented) have also been defined below.

1. How can an ontology-based translation mechanism be created for the purpose of automatic translation between different geo-analytical software environments?
    a. What are the technical elements necessary for ontology-based GIS workflow translation between geo-analytical software environments?
    b. How can common workflow goals be best expressed to produce GIS workflows across a range of different environments?
2. What is the capacity of the semantic data type signatures ontology (CCDT) for describing tools across geo-analytical software environments?

3. What is the quality of the GIS workflows generated through the implementation of the automatic translation mechanism?
    a. What assessment methods, if any, are available for GIS workflow translations, particularly within the GIS domain?
    b. What type of errors can be made in the workflows resulting from the translation of workflows between geo-analytical software environments?
4. What workflow similarity assessment methods are available for assessing similarity between GIS workflows translated between different geo-analytical software environments?
    a. What similarity assessment methods exist for the assessment of semantic and structural similarity between GIS workflows across different geo-analytical environments?
    b. What similarity assessment methods exist for GIS tool comparison across different geo-analytical software environments?

3.3 Research Limitations

Whilst this research thesis endeavours to assess the feasibility of translating workflows between geo-analytical software environments using and slightly adapting a semantic workflow synthesis mechanism, the time constraints of this research project place limits on the development of certain technical elements of the mechanism. Indeed, because the manual generation of the workflows, the specification and annotation of tools and the specification of the tasks is a time consuming process, the extent to which this is done will be limited to being sufficient for demonstrating and evaluating the implementation of the automatic translation mechanism and the ontology used for this purpose. It will also not be feasible for this research thesis to implement this translation mechanism across more geo-analytical software environments due to the time intensity of the research stages.

Chapter 3 Theoretical Framework

The function of this chapter is to situate the research being carried out within its wider scientific context. To do so, this chapter will be divided into distinct sub-sections, each of which presents the theory to support a research stage outlined in Chapter 4. The first subsection will provide the theoretical background for the development and use of processes and workflows in a range of contexts before discussing the use of workflow management systems in a scientific context. The next two subsections will discuss the technicalities of GIS workflow synthesis and translation using semantic technologies with the aim of identifying the technical elements which will need to be (re)produced as part of this project. These subsections will also introduce the core concept data types ontology (CCDT) (Scheider et al., 2020) used. The final two subsections will provide the theoretical frameworks for evaluating quality of the translated workflows, the functional capacity of the ontology, the synthesis engine, and the translation mechanism as well as assessment of similarity workflows from different software environments. These frameworks will provide the theoretical foundation for developing a method of quality and similarity assessment based on which conclusions on the effectiveness of the translation mechanism and preliminary comments on the similarity of software environments can be made. In situating this research within its wider scientific context, this chapter will provide a solid foundation for the production of a research methodology, aid in the evaluation of and reflection on the use of semantic technologies for the purpose of GIS workflow translation as well as provide justification for the research conducted in this thesis.

3.1 Workflows

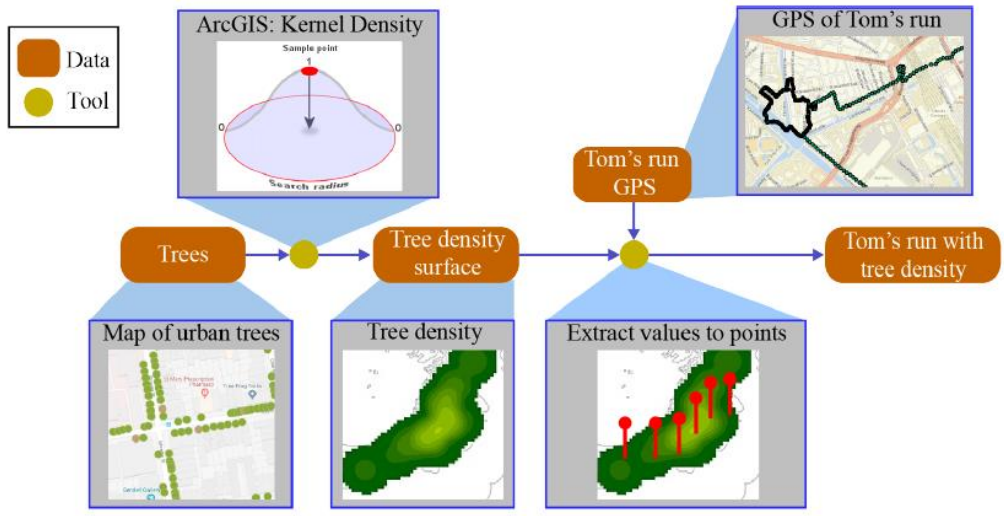
Workflows, in being defined as a 'sequence of steps involved in moving from the beginning to the end of a working process' (Merriam-Webster's Collegiate Dictionary, n.d.), have been a part of society since humans actively participated in work activities to produce an outcome or product as a result of following a set of concurrent steps. In their formal capacity, however, workflows find their modern origins in the early 20th century with the rise of business process automation where they were applied to the US manufacturing industry by Taylor and Gantt (Wilson, 2003) with the aim of improving business efficiency. Over the last several years, there has been a rise in the use of human computational algorithms or workflows which support the use of the crowd to tackle complex societal issues by breaking complex activities into manageable, self-contained tasks (Bernstein et al., 2010; Noronha et al., 2011; Kittur et al., 2011). Because the management of interconnected processes or activities through workflow creation was initially applied to business management practices, the body of research related to workflows has remained strongly associated with improvements in this field (Ludaescher et al., 2009). In recent decades, however, these management techniques have been the subject of intensive study; particularly as these techniques have begun to be applied to other fields (Leymann and Roller, 1997; Wilson, 2003; Barga & Gannon, 2007; Ludaescher et al., 2009; Gonzalez et al., 2010).

3.1.1 Scientific Workflows

Scientific workflows are defined a 'formal specification of a scientific process which represents, streamlines and automates the steps from dataset selection and integration, computation and analysis to final data product presentation and visualisation' (Lin et al., 2008). The development and the proliferation of the use of scientific workflows in a range of scientific fields is closely related to the rapid development of increasingly complex experimental and data analysis methods, computer simulations related to the

advent of e-Science research methods as well as potential for high volumes of data generation in any given area (Ludaescher et al., 2009). Indeed, while e-Science has provided new opportunities for increasingly data-driven and computationally intensive analyses in a range of research areas, this has also presented the scientific community with challenges relating to the management of large amounts of data and the navigation of increasingly complex computational environments when carrying out analysis. Because scientific workflows focus on automating information and computational processes in order to limit human error, and because these workflow tend to be well-specified and repeatable in line with general scientific practice, scientific workflows are well-positioned to aid scientists in carrying out complex and data-intensive analyses. As a result, there has been increased academic interest in scientific workflows composition and storage (Fox & Gannon, 2006; Gil et al., 2007; Ludaescher et al., 2009) as well as the proliferation of scientific workflow management technologies (Deelman et al., 2005; Fahringer et al., 2005). Whilst there are no established scientific workflow categories due to the lack of a unique set of characteristic features, Ludaescher et al. (2009) suggest that scientific workflows can be organised along a number of dimensions depending on the maturity of the workflow or the types of processes and operations being carried out as the workflow is implemented. An example of a scientific workflow is illustrated in the Figure below (Figure 3.1).

Figure 3.1. Scientific Workflow Example for Geospatial Analysis (Kruiger et al., 2020)



3.1.2 Scientific Workflow Management Systems

Whilst it is possible to generate scientific workflows by simply combining individual processes in a linear fashion, in reality, workflows are not generally made up of linearly connected processes; making scientific workflows complex to construct unaided and problematic when an unaided attempt is made due to the high potential for errors in the workflows. Whilst ad hoc scripting is often used by scientists in order to handle workflow construction and iteration, this scripting still requires that a researcher has sufficient knowledge of a scripting language, a background knowledge of the data being used in analysis, the workflow execution environment as well as in-depth knowledge of the processes and operations involved in the scientific workflow. A lack of said knowledge may place limitations on the quality of the workflows created (Barga & Gannon, 2009). Additionally, unaided workflow construction and ad hoc scripting may also limit the

reuse of scientific workflows in future research because secondary and tertiary researchers may not understand the scripting or scripting environments, the error conditions or have sufficient knowledge to repair the scripts where errors have occurred. Because reproducibility is a requirement of scientific research, unaided scientific workflow construction can, therefore, be problematic (Garijo et al., 2014).

To support the construction of robust, error-free scientific workflows, Scientific Workflow Management Systems (SWfMs) are developed and can be defined as a system which supports the modification, run, re-run and monitoring of scientific workflows to improve clarity around the availability of tool resources, simplify workflow construction and allow for the storing and reuse of workflows in line with scientific practices (Lin et al., 2008; Lamprecht, Margaria & Steffen, 2009; Kasalica & Lamprecht, 2018). These systems support the robust creation of scientific workflows in several ways. Firstly, SWfMSs often make use of a graphical user interface (GUI) where human actors can easily set task interdependencies, parameters and error conditions and visualise the workflow in an easily digestible manner (Chun et al., 2002). These GUIs make workflow construction far more accessible to individuals with limited scripting or process-dependent knowledge and allow for secondary and tertiary researchers to easily understand and edit the workflows. Secondly, these systems are often specialised to cater for the needs of a particular scientific domain or software environment (see, for example, Apache Taverna[1] for workflow management in the fields of bioinformatics, astronomy, and biodiversity). Finally, the storing of workflows in a digital format by the SWfMS is integral to the re-use, evaluation and editing by secondary and tertiary researchers. This need to store workflows have led to the rise of workflow repositories where workflows can be retrieved using queries specifying certain goals and inclusions necessary. These repositories then highlight relevant workflows which can be adapted to another context (Bergmann & Gil, 2014).

3.1.3 Limitations of Scientific Workflow Management Systems

Whilst the development and usage of SWfMSs in a range of scientific fields has been essential in supporting the construction, use and re-use of scientific workflows in various contexts, these systems do have a number of limitations. Firstly, whilst GUIs and domain-specific systems assist in reducing the knowledge barriers required for comprehensive scientific workflow construction, the construction of workflows in itself is not automated and, therefore, still requires a certain level of background knowledge of the data inputs, analytical operations and processes which make up a given scientific workflow (Zhang, Horvits & Parkes, 2013; Kasalica & Lamprecht, 2018; Scheider & Tomko, 2016). This level of expertise still places knowledge barriers on workflow construction, particularly where workflows are complex, where large databases are being accessed or where the tools involved in the workflows are domain specific. A solution presented in the literature is the increased sharing and reuse of scientific workflows. This sharing and reuse of scientific workflows, however, highlights further limitations of SWfMSs.

The sharing and reuse of scientific workflows can happen in three ways, namely; personal reuse by the initial creator of the scientific workflow, reuse of the scientific workflow by a collaborator involved in the initial construction and the re-use of the scientific workflow by a third party not involved in the original construction (Goderis et al., 2005). SWfMSs arguably do not pose any further limitation to the re-use of workflows in the case of personal reuse and collaborator reuse because workflows are easily shared, original environments generally assumed to be understood and communication between researchers easy. In the

[1] https://taverna.incubator.apache.org/

case of third-party re-use, however, SWfMSs present further limitations. Firstly, the SWfMS environment naturally dictates the naming conventions of tools, data movement, transformation, analysis, and visualisation used in the workflow (Deelman et al., 2009). This creates a silo of functionality within software environments and, therefore, a current lack of interoperability between different these.

Where third parties require the use of different SWfMSs than those used in the original workflow construction, this lack of interoperability makes the use of workflows across different systems difficult without extensive knowledge of both software environments, data sources, tools and processes (Ubels, 2018). Where rudimentary translation frameworks are available, some process details are lost in the translation due to a lack of capability (Oliveira, de Oliveira & Braganholo, 2015). Secondly, where workflows are shared for third party use, a lack of searchability of workflow specifications remains influential in limiting the reuse of scientific workflows. Several projects have been initiated to improve the searchability of workflows with limited success (Ubels, 2018). One of the greatest challenges for these projects is a formalised method of workflow description to remove ambiguity around processes and data. To date, this still requires a level of expertise and, therefore, poses a knowledge barrier to workflow reuse (Garijo et al., 2012).

Because the sharing of workflows for re-use does not offer a complete solution to improving the accessibility of comprehensive workflow specification and construction for non-experts and experts alike, SWfMSs have turned to the research and development of assisted or (semi-)automated workflow synthesis mechanisms. Here, workflow abstraction and semantic technologies reduce the knowledge barrier to construction comprehensive and error-free workflows in increasingly complex environments. These mechanisms and their related technical elements will be explored in the following subsection of this chapter, with a particular focus on scientific workflow synthesis in the GIS field in line with the focus of this thesis.

3.1.4 GIS Workflows, Scientific Workflows and Scientific Workflow Management Systems

A GIS is defined as ‘a system designed to capture, store, analyse, manage and present spatial or geographical data’ which allows users to ‘create interactive queries, analyse spatial data information, edit in maps and present the results’ as required (Clarke, 1986). Because a GIS uses data sources, information transformations and analytical operations when carrying out spatial data analysis, workflow management systems are well suited to supporting said analysis; particularly where there are large amounts of data involved. This is done by offering user-friendly interfaces for building workflows and in-built constraints to limit errors based on invalid combinations of tool types. This reduction in complexity helps to improve the quality and accessibility of these analyses by inexperienced or non-experts as well as the range of contexts in which GIS analysis can be applied. Unfortunately, however, despite the presence of workflow management systems, the growing number of tools, data sources and valid combinations of tools for a single analytical task is challenging to keep familiar with and is further complicated by the fact that there are multiple software environments available for carrying out said computations. Analysts and non-experts alike unable to stay on top of this computational complexity cannot fully exploit the available resources and potentially limits the complexity or comprehensiveness of the analysis being carried out.

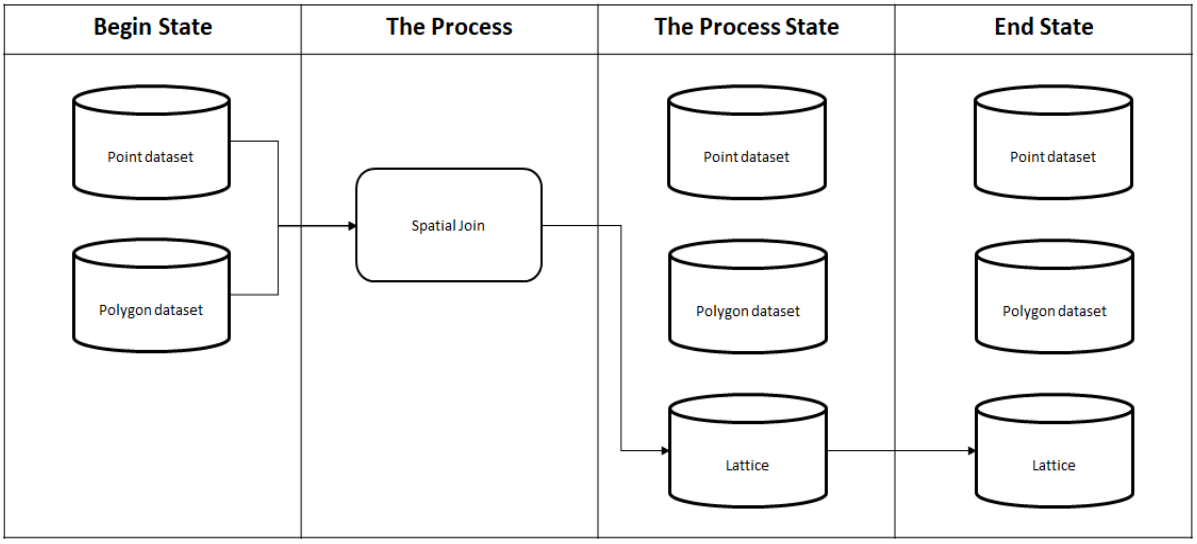
The literature on GIS workflow composition focusing on reducing complexity has turned to the exploration of assisted or (semi-)automatic workflow synthesis mechanisms supported by abstraction methodologies and semantic technologies. Within the field of GIS, the literature has suggested the use of semantics-based

geo-service descriptions to achieve computational goals (Hofer et al., 2017) or the use of geo-operator thesauri and pairwise service matching for retrieving tools for a given purpose (Brauner, 2015); both techniques have been implemented with varying levels of success for a variety of reasons. The idea of loose programming is often suggested as a useful conceptualisation for this purpose as this does not require the use of procedural code, but rather the underspecification of analytical processes, use of computational goals and constraints to develop algorithms for automatic workflow generation (Kruiger et al., 2020).

3.2 Workflow Synthesis

Despite the presence of SWfMSs, these computational environments remain fairly complex and still require a level of understanding of the data sources, the processes, operations and software environments. A potential solution to this issue is the development and use of an automatic workflow synthesis mechanism which is able to generate valid and sufficiently complex workflows able to implement analysis based on certain goal specifications (Zhang, Horvits & Parkes, 2013; Kasalica & Lamprecht, 2018). The use of these synthesis mechanisms changes the focus of SWfMSs from assisting users with the manual construction of scientific workflows to managing the automatic generation and validation of workflows based on given objectives (Chun et al., 2002; Martin et al., 2005; Kasalica & Lamprecht, 2018). Workflow synthesis mechanisms, in general, are based on workflow abstraction through the concept of loose specification, defined as 'a specification which features elements of underspecification', allowing the creator of a workflow to omit certain details in describing the goals of specific tasks in a workflow (Lamprecht, Margaria and Steffen, 2010). This loose specification is a form of exploratory scientific workflows wherein the full specification of goals is left to later stages of the development of a specific workflow. Loose specification is done in 4 stages, namely, begin state, the process, the process state, and the end state. The following figure (Figure 3.2) illustrated these states for a simple GIS workflow.

Figure 3.2. Specification of a Simple GIS Workflow



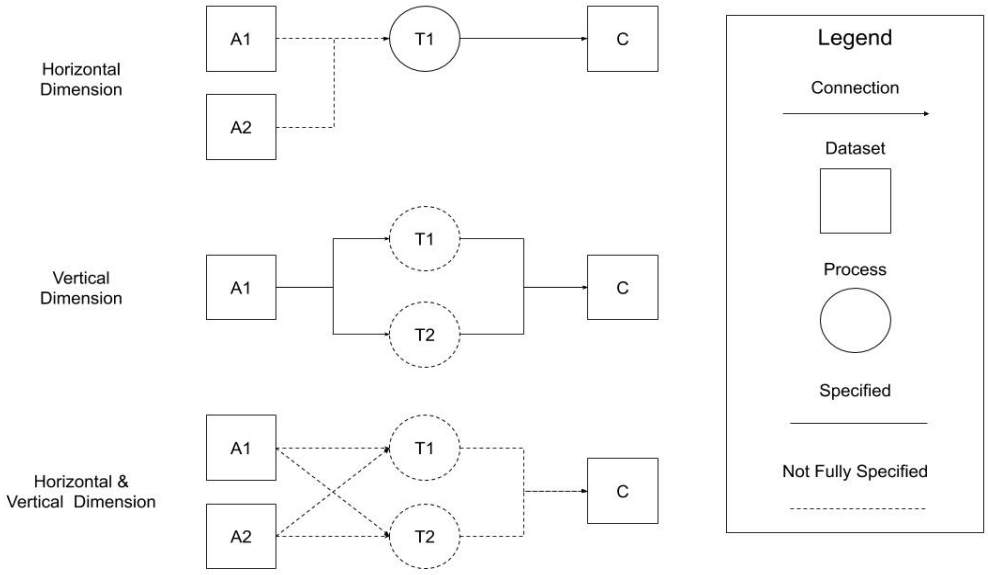
As illustrated by the above figure, the begin state is a specification of the starting state of the system being used. In the case of a GIS, this state describes the data available as input for the analysis being carried out based on a specific analytical question. In the illustration above, the begin state is made up of two datasets,

a point dataset commonly used to represent non-adjacent features with zero dimensions as well as a polygon dataset representing spatial areas with discrete boundaries such as, for example, a lake or municipality boundaries. The process is the implementation of a specific analysis which produces a process state or new system state following the completion of said analysis. In the illustration of a GIS analysis above, the process state would be the implementation of a spatial join function which affixes data from one feature layer's attribute table to another feature layer based on the spatial overlapping of these features. The new system resultant from this process state, in this case a lattice broadly defined as a surface using an array of regularly spaced sample points, is the input for the end state which then produces the workflow output. The output in this example is also a lattice due to the computation constraints placed on each tool. Whilst this basic form of loose specification certainly assists with reducing the complexity of the computational environment, this still requires a level of workflow or domain-specific knowledge as well as potentially some scripting knowledge to specify certain states. As such, this loose specification remains a form of assisted workflow synthesis rather than a form of complete automatic workflow synthesis.

Loose programming, by extension, 'aims at an environment where such details [states] can be added automatically according to given semantic constraints' or semantic properties (Lamprecht et al., 2010). Here, functions in procedural code are not explicitly called but, rather, a user is able to feed computational goals and constraints into a given algorithm with the goal of generating workflows with specified lengths satisfying these constraints (Kruiger et al., 2020). These constraints are expressed as semantic representations of data inputs (begin states) and tool specifications are placed in concept hierarchies which the algorithm then reasoned over to produce valid workflows. This form of programming has recently been applied, albeit infrequently, to geographic information processing with a degree of success (Scheider et al., 2016; Kasalica and Lamprecht, 2019). This form of programming allows for the underspecification of some or all of the above states along two dimensions, namely, the horizontal and the vertical dimension (Lamprecht et al., 2010).

The horizontal dimension sees the abstraction of the workflow through underspecification of the connections between states and processes, allowing the creator of the workflow to only have to specify the inputs, outputs, and the processes. The vertical dimension sees the underspecification of processes, allowing the creator of the workflow to specify the inputs and connections. Importantly, it is also possible to combine both horizontal and vertical underspecification to specify a workflow without fully specifying any connections or processes (Lamprecht et al., 2010). The types of underspecification along certain dimensions are illustrated in the figure below (Figure 3.3). It is the combination of the vertical and horizontal underspecification in this way, in combination with the presence of specified semantic constraints supported by formalised ontologies, which makes automatic workflow synthesis for GIS workflows possible. Here, only begin and end state of the workflow need to be known and the user of the synthesis mechanism need only to have task specifications, based on data and task constraints, for each analytical question for which a workflow needs to be constructed. Task specifications are included in the competency questions defined at the beginning of an analysis, where the begin and end states of a workflow in terms of its data states are semantically defined (see section 3.2.4).

Figure 3.3. Horizontal and vertical loose specification (adapted from Lamprecht et al., 2010).



In order to automate the process of workflow construction, automatic workflow synthesis mechanisms need to make use of the synthesis methodology which, in turn, makes use of loose specification in both the horizontal and vertical dimensions (Steffen, Margaria & Freitag, 1993; Freitag et al., 1995). As discussed by Steffen et al. (1993), workflow synthesis using this methodology makes use of modal logical semantic linear time logic (SLTL) which is a formal language for specifying queries over workflows in terms of relevant semantic hierarchies and combines taxonomic classifications of the relevant data types and processes in a workflow with relative time. In doing so, SLTL combines three types of constraints together which interprets a given task specification for a workflow and automatically generates all process combinations that satisfy this specification and, therefore, producing a number of valid workflows. As such, the synthesis of workflows is a form of information retrieval wherein workflows are generated based on the querying of a specific (tool or data) taxonomy to achieve an answer based on specified constraints. The three types of constraints used in a synthesis algorithm are static constraints, dynamic constraints, and temporal constraints (Lamprecht et al., 2010). Static constraints are characterised by taxonomic expressions over data types or classes denoting which data types match which process or classes of processes in a workflow. Dynamic constraints are the taxonomic expressions over the processes or classes of processes in a workflow and temporal constraints are taxonomic expressions which are covered by the modal structure of logic. To successfully implement an automatic workflow synthesis mechanism, loose specification of workflow elements and the synthesis methodology need to be supported by technical inputs based on semantic technologies. This includes a formalised ontology in the relevant knowledge domain as well as annotated tool and data databases based on this formalised ontology.

Lending methods from the field of information retrieval, workflow synthesis in this way can consider a given problem either locally or globally. Where a problem is specified as being local, the implemented synthesis algorithm will take as input only the output of the process immediately preceding it when formulating the appropriate process connections in a workflow. When a problem is approached by the

synthesis algorithm as being global, the algorithm will seek to use all the resources within a given system in constructing appropriate process connections. The complexity of GIS workflows and analysis problems means that these tend to require a global approach to a synthesis problem. This should be specified where synthesis machines are applied to GIS workflow problems.

3.2.1 Competency Questions and Workflow Task Specification

In the most fundamental sense, workflows in the GIS field are implemented to answer geo-analytical questions (Scheider et al., 2016). As such, it is important that the questions being asked and, where necessary, the outputs required are as specific as possible to produce meaningful analysis. In making use of automatic workflow synthesis mechanisms, therefore, it is important to first ensure that workflow goal specifications are as detailed as possible; particularly where mechanisms are implemented under complete underspecification contexts. To achieve this detail, workflow synthesis mechanism has made use of project competency questions in designing workflow goal specifications (Gangemi, 2005; Scheider & Tomko, 2016; Scheider & Ballatore, 2018). Competency questions are defined by Gangemi (2005) as a typical query submitted by an expert with the goal of accessing a certain knowledge base within a given domain for defined tasks and are commonly used in the creation and evaluation of ontology design patterns (Gangemi, 2005; Hitzler, Gangemi & Janowicz, 2016; Scheider et al., 2016). In constructing competency questions for workflows, it is necessary to specify the begin state in the form of dataset inputs and goal specifications which serve as inputs for the automatic workflow synthesis engine. It should be noted, however, that the goals specified as part of this process only indirectly provide the answers to the questions being asked with the intention of providing a way of validating whether a meaningful workflow has been produced through synthesis (Scheider, Ostermann & Adams, 2016). This is because the goal indicates an output type, such as for example a data type, rather than an output itself, such as a choropleth map. An example (Table 3.1) of a competency question and its respective goal specifications are provided below.

Table 3.1. Competency question example (adapted from Kruiger et al., 2020).

Question	What is the number of sports facilities in each PC4 area?
Given Data	Sports facilities are interpreted as objects and represented by point vectors with a nominal attribute denoting the facility type; PC4 areas in Amsterdam form a Vector Lattice.
Goal Specification	The goal is a Vector lattice on the PC4 level with extensive counts.

In specifying a goal for the implementation, the competency questions also serve to evaluate the automatic workflow synthesis mechanism and the underlying ontology by ensuring output types match the data types specified (Wieleman, 2018).

3.2.2 Information Science Ontologies

As discussed by Steffen et al. (1993), the workflow synthesis methodology makes use of SLTL to support the synthesis of workflows by making use of semantic hierarchies and taxonomic classifications when producing process combinations for valid workflow generation. As such, in order to create and successfully implement a workflow synthesis mechanism which is able to automatically add the elements required to

synthesise valid workflows from specified inputs and outputs, a machine-readable formalisation of a specific knowledge domain is required (Kasalica & Lamprecht, 2018; 2019). This formalisation is achieved through the taxonomic classification of domain knowledge known as a light ontology and is generally built using ontology development languages based on Extensible Mark-up Language (XML).

The most notable ontology development languages include Resource Description Framework (RDF) and the Ontology Web Language (OWL), the latter of which is the most expressive language (Munir & Sheraz Anjum, 2018) and has been used in the development of GIS-centred ontologies such as Scheider et al.'s CCDT ontology (2020). The reason why OWL is often used over RDF for the development of ontologies is due to the ability of the language to model implicit domain knowledge usually inherently understood by domain experts (such as spatial core concepts) as explicit knowledge through reasoners which enable class axioms, property restraints on certain classes and relations between properties. RDF, in contrast, is limited in its ability to set these sorts of constraints which model implicit knowledge and are, therefore, generally better for annotations which form part of semantic technologies.

Ontologies are generally built in ontology builders such as Protégé which is a preferred tool within the geospatial sciences (Albrecht, Derman & Ramasubramanian, 2008). An ontology within the information sciences is generally defined as a formal representation and naming of a conceptualisation (Zuniga, 2001); including categories, properties, and relations between a number of concepts, processes, data and individual entities (Gensesereth & Nilsson, 1987) in order to make the implicit knowledge within a given domain explicit (Gruber, 1995; Guarino, 1998). Literature discussions within the information science field on the formalisation of ontologies identified the following common components of ontologies (Table 3.2).

Table 3.2. Common Components of Ontologies in Information Science

Component	Definition
Individuals	Instances or objects within a specific knowledge domain
Classes	Sets or collections of instances of objects
Attributes	Properties or features of objects
Relations	The ways in which individuals and classes are related to each other
Function Terms	Complex structures formed from relations and used in place of individuals in a statement
Restrictions	Formally stated descriptions which must be true for an assertion to be accepted as input
Rules	If-then statements which describe logical inferences
Axioms	Assertions in a logical form which make up the overall theory underpinning the domain application which the ontology is describing
Events	Changes to attributes and relations

Compiled and adapted from Gruber (1995), Zuniga (2001), Taniar (2006), Gomez-Perez, Fernández-López & Corcho (2006), Donnelly & Guizzardi (2012).

There are four types of information ontologies, namely, top-level ontologies, domain ontologies, task ontologies and application ontologies, where the type of a given ontology depends on how generally the ontology describes the knowledge domain in question (Guarino, 1997). Top-level ontologies describe concepts very generally and tend to incorporate a number of knowledge domains. Domain and task level ontologies describe the vocabulary related to a particular knowledge domain or a particular activity. Application ontologies describe concepts related to both the task and the knowledge domain by describing the roles of entities in an activity related to a particular knowledge domain. These ontology types are not mutually exclusive of one another but rather are somewhat hierarchical. Indeed, the specialised terms used in domain and task ontologies will be derived from a relevant top-level ontology and, in turn, the application ontology will be derived by combining the specialised terms from domain and task ontologies. In the case of GIS workflow synthesis, the light ontology used would fall into the application ontology domain due to the fact that the mechanism requires machine-readable descriptions of data types (domain knowledge) and processes (tasks) in order to carry out the activity of producing outputs from inputs, where the data types serve as inputs and outputs for the processes.

Information science ontologies can be implemented with the role of providing the architecture for a given information system by using top-level or domain ontologies or can be used to implement activities within an information system itself by using task or application ontologies (Guarino, 1998). Whilst the application architecture is based on a formalised GIS domain ontology, the system does not actively check an ontology when carrying out tasks and activities. When this is done, the underlying ontology is an application ontology. Because GIS workflow synthesis requires that the underlying ontology be actively checked due to the underspecification of tasks and processes, the underlying ontology needs to be an application ontology. Unfortunately, these application ontologies are rare within information science fields and are not readily available for implementation in GIS applications (Guarino, 1998).

3.2.3 GIS Workflow Synthesis Ontologies

Whilst there has been some successful work on the use of semantic technologies for automatic chaining of geo-functionalities and -data, much of this work and the resulting ontologies have been focused on discovery, composition or execution of GIS operations and workflows in the context of service composition (Lemmens et al., 2006; Lutz, 2007; Yue et al., 2007; Fitzner et al., 2011). Indeed, because an essential component of an ontology is the capturing of semantic intricacies regarding domain-specific terminology, these ontologies are not useful for the purpose of workflow synthesis as they do not adequately capture the geo-analytical terminology surrounding GIS workflow synthesis (Hofer et al., 2017). To do this, extensive knowledge of the semantic terminology related to GIS and GIS workflows is needed (Scheider & Tomko, 2016). Currently, semantic validity is assessed based on relevant literature and theory or by using formal logic (Scheider & Tomko, 2016; Hofer et al., 2017). This method for ontology creation was implemented by Scheider et al. (2020) in the production of the CCDT ontology with workflow synthesis as a specific goal. It is argued that the core concepts of spatial information provide abstractions of GIS related knowledge which assist analysts in formulating analytical questions and compute the appropriate answers over geodata with different formats (Scheider et al., 2016). The ontology formulated from these core concepts has been subsequently evaluated for its use in GIS workflow synthesis by Kruiger et al. (2020).

a. Semantic Data Type Signatures Ontology (CCDT)

The Semantic (Core Concept) Data Type Signatures Ontology (CCDT) designed and developed by Scheider et al. (2020) is perhaps the first GIS-centred application ontology available. In previous work on the subject, Scheider et al. argue that abstract data types, more than software environment-specific GIS operations and tasks, are useful features to model for the purpose of GIS workflow synthesis as these data types constrain computational methods to the types of data that can be meaningfully run. This, therefore, prevents nonsensical computations while still allowing generalisability (2016). The abstract data types identified as useful for assisting with the synthesis of meaningful and valid GIS workflows are the core concepts of spatial information (Kuhn, 2012) and related geo-semantic (attribute) distinctions. These abstract data types are fundamentally a combination of geometric layer types, spatial core concepts and other special attribute types. It is noted that the core concepts of spatial information are not concrete data types and can still be interpreted in a variety of ways by different analysts across a variety of (GIS) software environments. As such, it is still necessary to produce semantic data types which can capture this variety (Scheider et al., 2016).

Geometric Layer Types for Representing Core Concepts

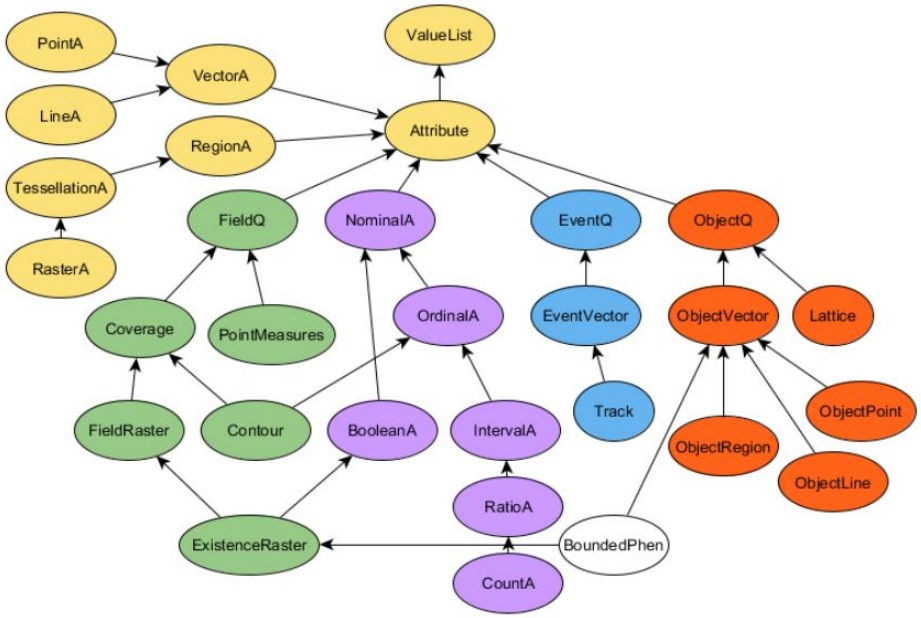
The first semantic distinction identified as relevant by Scheider et al. (2016) for knowledge formalisation is the geometric property of GIS layers, defined as those properties that can be obtained from the geometry of a solid object. Here, while GIS is often distinguished according to the two geodata types of raster and vector, this distinction for analysis is superfluous due to the fact that translations between formats is possible without really altering the underlying format of the data and that ignoring this distinction is often important for analysis. Instead, Scheider suggests a shift in focus from the distinction between layer types to a focus on the geometric properties of layers where a distinction should be drawn between 'layers that are Tessellations or not, as well as between point, line or region datasets' (2019). This distinction, it is argued, allows the representation of the core concepts of field. A reorganisation of geodata layers along these principles creates a hierarchy of types where raster data becomes a subtype of polygon vector data. It is based on this hierarchy that the description logic notation is derived and specific definitions and axioms describing the intricacies of the hierarchy provided (see Scheider, 2019).

Spatial Core Concepts and Attribute Semantics

The second group of semantics discussed by Scheider (2019) is the spatial core concepts and attribute semantics. The core concepts considered here were first introduced by Kuhn with the purpose of 'redesigning a novel abstract interface for GIS' (2012 as cited by Scheider et al., 2016). The core concepts are captured in values at different measurement scales; determining the type of mathematical operation which can be applied to the underlying data during analysis (Kuhn, 2012). To produce abstract data types, Scheider et al., focus on only four core concepts: fields, objects, networks and events (2016). Spatial core concepts need to account for the semantic properties of attributes, which includes the measurement levels that determine the type of numerical operation that can be applied to the attribute data (Chrisman, 2002). To capture this, each core concept has as set of attributes where an attributes is defined as a ValueList of a given SpatialDataset. This ValueList inherently has a data measurement level depending on the type of attribute data that it is describing and this dictates what mathematical operations can be executed in a meaningful way when analysis of spatial information is run over the data (Stevens, 1946).

To put this more concretely, a tessellated representation of a field (called coverage) can be used to represent a spatially continuous value surface with discreate tiles. This representation has a special property where 'parts of its regions have the same attribute value as the entire region' which allows for the reconstruction of the values from the continuous field by subdividing geometries. This property is an essential property of fields, and objects, in contrast, do not have said property. Scheider (2019) argues that this property distinction between fields and objects is not captured by defining objects as discrete while fields are not, since all mentioned data models are bound to be discrete in nature. The following figure (Figure 3.4) builds on the previous in providing an illustrative summary abstract types of spatial attributes, which are realised by combining the preceding geometric layer types, spatial core concepts and other attribute types to provide the basis for the CCDT Ontology.

Figure 3.4. Abstract types of spatial attributes, combining geometric layer types, spatial core concepts and measurement levels (Scheider, 2019).



The table below (Table 3.3) defines the possible measurement levels that spatial attributes can take on and provide an example of this attribute in practice.

Table 3.3. Attribute Type Definitions and Axioms (Kruiger et al., 2020)

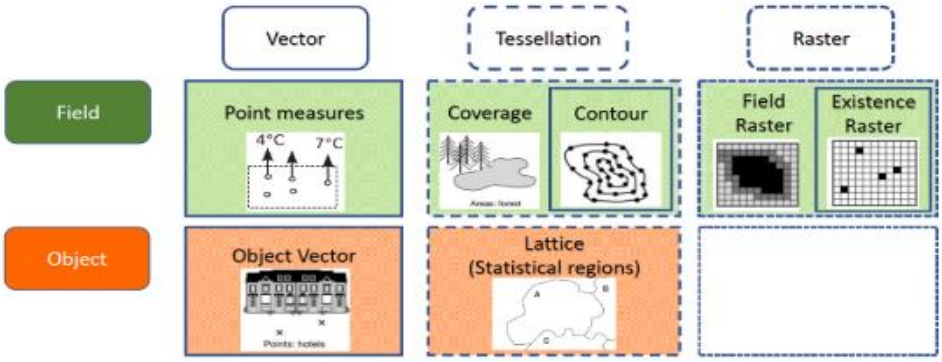
Attribute Definition/Axiom	Explanation	Example
Attribute	Any attribute of a layer	Population (of a layer with cities)
NominalA⊆Attribute	Attribute on a nominal scale (can be compared for identity)	Land use type
BooleanA⊆NominalA	Attribute on a nominal scale with only two possible values	Is policy in effect here?
OrdinalA⊆NominalA	Attribute on an ordinal scale (can be compared for less/equal than as well)	Disease intensity of a tree

IntervalA⊆OrdinalA	Attribute on an interval scale (a difference is meaningful)	Temperature (in degrees Celsius)
RatioA⊆IntervalA	Attribute on an interval scale (ratios and 0 are meaningful)	Temperature (in Kelvin)
CountA⊆RatioA	Attribute that represents a count of discrete entities	Number of sport facilities

Synopsis of Abstract Data Types

The abstract types of spatial attributes, when freely combined, result in a total of 72 different data type signatures, not all of which are valid or produce meaningful analysis. Scheider (2019) combined these signatures into 6 categories relevant and meaningful for geographic operations concerning fields, objects, networks and events; the latter two of which is not relevant for this thesis. The following definitions and axioms make up the CCDT ontology pattern (Scheider et al., 2016) which, in combination with the preceding discussions, is used as the basis of the CCDT ontology (Scheider, 2019; Scheider et al., 2020). The definitions and axioms in the synopsis are sorted according to their core concept types, namely, field-based types and object-based types as illustrated by the following figure (Figure 3.6).

Figure 3.5. Adapted from matrix of data types based on combining geometric layer types with spatial core concepts (Scheider, 2019).



Field-Based Types

Definition 6. PointMeasures ≡ FieldQ∩∃ofDataSet.PointDataSet

This definition denotes the argument that point measures are the simplest way to represent a field. The underlying dataset here is a vector dataset as a result of Axiom 2 and, therefore, the dataset cannot be used to estimate the value of the field quality at any given location in the field.

Definition 7. FieldRaster ≡ FieldQ∩∃ofDataSet.Raster

This definition denotes a typical field representation based on tessellation principles. This can be used to infer the underlying field quality at any given location within the range of the data in question.

Definition 8. Coverage ≡ FieldQ∩∃ofDataSet.Tessellation

This definition represents a more generalised way of representing a field because a tessellation based on field values rather than a strict raster dataset is the underlying data input. This allows coverage to have self-similar properties as discussed above.

Axiom 6. Contour⊆Coverage∩OrdinalA
This axiom denotes the argument that contours are a particular type of coverage since it is ordinally scaled through quality intervals.

Definition 9. ExistenceRaster ≡ FieldRaster∩BooleanA
An existence raster is a special subtype of field representation wherein the field raster has boolean value ranges. This type of raster includes all the spatial core concepts for each location and denotes a true value where phenomena appear. An example of this existence raster is a land-use dataset wherein true is a value of a cell where land use is found.

Definition 10. ExistenceVector ≡ FieldVector∩BooleanA
This definition makes use of the same logic as the existence raster; however, existence vectors are Boolean Vector Fields which represent the existence of irregular geometries or 'patches' of a phenomenon in question at a particular location.

Object-Based Types

In contrast to field-based types of attributes, object-based attribute types are not self-similar, and it cannot be assumed that parts of their geometries have the same attribute values as the whole object. As such, the following definitions are relevant.

Definition 11. ObjectVector ≡ ObjectQ∩∃ofDataSet.Vector
This definition denotes the existence of an object as a vector such as a building or point of interest in a large dataset. These datasets are likely to have spatial gaps were no object or object quality is located. Objects have boundaries which are meaningful and countable. This leads to the generation of a bounded phenomenon supertype in the following definition.

Definition 12. Lattice ≡ ObjectQ∩∃ofDataSet.Tessellation
This definition denotes the argument that a lattice is a tessellated representation of an object and its quality, therefore implying that object boundaries form a tessellation where there are no spatial gaps in the data. An example of this is statistical regions containing statistical data about population.

The formalisation of the ontology in this way allows for GIS tools and data inputs and outputs for given workflows to be annotated accordingly and allows for the development of annotated databases of tools which then form part of a configuration and implementation of the workflow synthesis mechanism.

3.2.4 Annotated Database Creation

In line with previous work on workflow synthesis within the geospatial sciences (Kasalica & Lamprecht, 2018; Meerlo, 2019; Kruiger et al., 2020), a full formalisation of a domain model with the goal of implementing an automatic workflow synthesis mechanism requires that annotated databases of data, tools and processes as necessary need to be developed. These databases should be annotated in accordance with

the relevant ontology (Gruber, 1995; Lamprecht et al., 2009; Kasalica & Lamprecht, 2018) and provide machine-readable descriptions of GIS tools, data and processes in line with semantic web technologies and linked data principles (Berners-Lee & Hendler, 2001). The practical execution of these annotations is done in a structured manner according to RDF principles. Developing annotated databases in this way ensures that the information within the database is explicitly defined and linked to other (external) datasets (Bizer, Heath & Berners-Lee, 2009).

b. Resource Description Framework (RDF)

Perhaps the common framework, used to structure linked data in annotated databases is the Resource Description Framework which is a data model representing a given semantic network. Where OWL is generally more popular for the purpose of ontology design and development, RDF, and its sub-formats such as Turtle, is more widely used for resource description such as annotated database creation. As discussed by Bizer et al. (2009), the framework is made up of a directed network wherein qualified edges specify a relation between two objects in a certain direction. These relations are heterogeneous in nature given that every edge can have a different meaning. Each object in an RDF network is described by a Unique Resource Identifier (URI) which is similar in structure to a Uniform Resource Locator (URL) and denotes the domain, RDF namespace or the publisher of the object in the semantic web (first half of the URI) as well as the unique resource for the object (the latter half of the URI). The directed nature of the network is created by using RDF triples which are made up of three parts, namely, the subject, predicate, and object. The subject and object are denoted by URI as the predicate describes the relationship between these two parts. Naturally, triples do not exist in isolation but instead connect to one another to form a graph-like information structure. This allows information to be discovered about a particular subject by simply following the relationships between the nodes in the graph.

RDF databases of GIS tools need to be serialised to make it machine-readable for the purpose of workflow synthesis. There are two methods which can be used to carry out this serialisation, namely, RDF/XML serialisation and Turtle serialisation. RDF/XML serialisation converts the triples shown above into resource, property, and property value. Here, a resource is anything which can be represented by URI or a node in a graph. The property is the relationship or predicate between two nodes and the property value is the object part of the triple. What distinguishes the resource from the property value is the direction of the property or predicate between these. This method of serialisation follows a strict structure wherein an XML declaration is made, and root elements are used allowing for the quicker description of URI domains and shorter overall formalisation. The rdf:about element is repeatedly used to define the resource in every new triple and <rdf:Description> is constantly used to open and close a description of a resource.

Figure 3.6. XML Serialisation of part of the CCDT Ontology

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://geographicknowledge.de/vocab/CoreConceptData.rdf#"
    xml:base="http://geographicknowledge.de/vocab/CoreConceptData.rdf"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xml="http://www.w3.org/XML/1998/namespace"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <owl:Ontology rdf:about="http://geographicknowledge.de/vocab/CoreConceptData.rdf">
```

```
        <rdfs:comment xml:lang="en">This ontology pattern describes geodata types representing core concepts of
spatial information (Kuhn 2012) we well as further semantic attribute distinctions relevant for geocomputation
and geo-analysis.</rdfs:comment>
        <rdfs:seeAlso rdf:resource="http://spatial.ucsb.edu/core-concepts-of-spatial-information/"/>
    </owl:Ontology>
    <!--
    ///////////////////////////////////////////////////////////////////////////////////
    // Object Properties
    ///////////////////////////////////////////////////////////////////////////////////
     -->
    <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#hasElement -->
    <owl:ObjectProperty rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#hasElement"/>
     <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#ofDataSet -->
    <owl:ObjectProperty rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#ofDataSet"/>
        <!-- http://geographicknowledge.de/vocab/GISConcepts.rdf#hasAttribute -->
    <owl:ObjectProperty rdf:about="http://geographicknowledge.de/vocab/GISConcepts.rdf#hasAttribute"/>
        <!--
    ///////////////////////////////////////////////////////////////////////////////////
    // Classes
    ///////////////////////////////////////////////////////////////////////////////////
     -->
    <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#Attribute -->
    <owl:Class rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#Attribute">
        <owl:equivalentClass
rdf:resource="http://geographicknowledge.de/vocab/CoreConceptData.rdf#Attribute"/>
    </owl:Class>
        <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#Spatial -->
    <owl:Class rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#Spatial"/>
        <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#SpatialData -->
    <owl:Class rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#SpatialData"/>
        <!-- http://geographicknowledge.de/vocab/AnalysisData.rdf#SpatialDataSet -->
    <owl:Class rdf:about="http://geographicknowledge.de/vocab/AnalysisData.rdf#SpatialDataSet"/>
```

Turtle serialisation uses the same resource, property and property value serialisation principles but has made slight improvements in the way elements are written. Indeed, this method of serialisation uses @prefix to define root elements which allow for the usage of prefixes in place of namespaces rather than whole URIs. Turtle also does not require the use of rdf:about to specify a resource as the resource and its description is specified based on the URI used in the first row of the indentation. The use of indentation in this way also allows for extra property and property value resources to be added to the resource without having to redefine the resource as in RDF/XML serialisation. This method of serialisation is presented in the figure below (Figure 3.7).

Figure 3.7. An Example of Turtle Serialisation for ArcGIS tools

```
@prefix wf: <http://geographicknowledge.de/vocab/Workflow.rdf#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix tools: <http://geographicknowledge.de/vocab/GISTools.rdf#>.
@prefix ccd: <http://geographicknowledge.de/vocab/CoreConceptData.rdf#>.
@prefix exm: <http://geographicknowledge.de/vocab/ExtensiveMeasures.rdf#>.

<https://desktop.arcgis.com/en/arcmap/latest/extensions/geostatistical-analyst/using-areal-interpolation-to-
```

```
predict-to-new-polygons.htm> tools:implements tools:ArealInterpolationAverage.
tools:ArealInterpolationAverage rdfs:label "Makes polygon to polygon predictions of the mean of a measured
attribute";
   wf:input1 [ a ccd:Lattice, ccd:VectorA, ccd:IntervalA, exm:IRA ];
   wf:input2 [ a ccd:Lattice, ccd:VectorA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:IntervalA, exm:IRA ].

<https://desktop.arcgis.com/en/arcmap/latest/extensions/geostatistical-analyst/using-areal-interpolation-to-
predict-to-new-polygons.htm> tools:implements tools:ArealInterpolationRate.
tools:ArealInterpolationRate rdfs:label "Makes polygon to polygon predictions of a measured rate attribute";
   wf:input1 [ a ccd:Lattice, ccd:VectorA, ccd:RatioA, exm:IRA ];
   wf:input2 [ a ccd:Lattice, ccd:VectorA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:RatioA, exm:IRA ].

######### Spatial join with sum rule
# with Join One to One parameter, with Merge Rule Sum for Ratio data.
<https://pro.arcgis.com/en/pro-app/tool-reference/analysis/spatial-
join.htm> tools:implements tools:SpatialJoinSumTessRatio.
tools:SpatialJoinSumTessRatio rdfs:label "Sums the attributes at ratio measurement level from one feature to
another based on the spatial relation";
   wf:input1 [ a ccd:ObjectVector, ccd:RatioA, exm:ERA ];
   wf:input2 [ a ccd:Lattice, ccd:VectorA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:RatioA, exm:ERA ].

# with Join One to One parameter, with Merge Rule Sum for Count data.
<https://pro.arcgis.com/en/pro-app/tool-reference/analysis/spatial-
join.htm> tools:implements tools:SpatialJoinSumTessCount.
tools:SpatialJoinSumTessCount rdfs:label "Sums the attributes at Count measurement level from one feature to
another based on the spatial relation";
   wf:input1 [ a ccd:ObjectVector, ccd:CountA, exm:ERA ];
   wf:input2 [ a ccd:Lattice, ccd:VectorA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:CountA, exm:ERA ].
```

These subtle changes improve the readability and writability of the RDF file but do slightly weaken the parsability of the file in comparison to XML serialisation. The method chosen for annotated tool database creation depends on analyst preference, program requirements and whether databases are being manually generated. Either method of serialisation will be useful in creating the annotated databases required for GIS workflow synthesis as illustrated by Kasalica and Lamprecht (2018) and Kruiger et al. (2020).

3.2.5 GIS Taxonomy Preparation

Following the generation of the annotated tool databases, it is necessary to create a taxonomy of types (Kruiger et al., 2020). This taxonomy is a Resource Description Framework Schema (RDFS) hierarchy (consisting of rdfs:subClassOf triples) of tools and data classes created based on the tool annotations and the formalised CCDT ontology. This taxonomy is created in preparation for the configuration and implementation of the Advanced Pipeline Engine (APE) where the GIS tool annotations, the goal and task specifications and the GIS taxonomy are fed into the engine (see following section). This taxonomy is created and altered as necessary using the relevant Python script, the basis of which is an OWL reasoning

step which infers statements regarding class combinations occurring in a tool annotation. This OWL reasoning and the development of the Python script is beyond the scope of this thesis and has instead been provided by an expert. This research does, however, make use of this script to alter the taxonomy when alterations are made to the tool annotations.

3.2.6 Automatic Synthesis Engine

The automation of GIS workflow synthesis has been explored with the development of workflow synthesis engines based on the synthesis methodology and SLTL. These engine use as configuration three inputs, namely, the GIS taxonomy produced from the CCDT ontology and the annotated databases, the annotated databases themselves as well as task specifications. These task specifications consist of the analytical goals or goal specification for each workflow (end states) and the input for each workflow in the form of semantically described datasets. Indeed, these specifications are machine readable formats of the natural language inputs and goal specifications defined by the competency questions and serve to loosely specify a given workflow in both the horizontal and vertical dimensions. In recent work on GIS workflow synthesis, the CCDT ontology has been used to provide the semantic properties of workflow inputs (datasets) and goal specifications in terms of their abstract data types (Kasalica & Lamprecht, 2018; Kruiger et al., 2020). An example of a task specification is illustrated in the table which follows (Table 3.4).

Table 3.4. Task Specification Example (adapted from Kruiger et al., 2020).

Competency Question	What is the number of sports facilities in each PC4 area?
Given Data	Sports facilities are interpreted as objects and represented by point vectors with a nominal attribute denoting the facility type; PC4 areas in Amsterdam form a Vector Lattice.
Input specification	ObjectPoint∩NominalA; Lattice∩VectorA
Goal Specification	The goal is a Vector lattice on the PC4 level with extensive counts.
Goal Specification (semantic properties)	CountA∩VectorA∩Lattice

A synthesis engine by the name of APE (Kasalica & Lamprecht, 2020a, b) has been used in preliminary work on automatic GIS workflow synthesis by Meerlo (2019), Kasalica and Lamprecht (2019) and Kruiger et al. (2020). APE is a Java supported engine and developed for the purpose of GIS workflow synthesis. While the current state of the engine does not offer a user-friendly graphical interface, it does have the ability to limit workflow patterns based on predefined restriction and constraint inputs. These are, however, beyond the scope of this research thesis. The GIS taxonomy (serialised as RDF/XML format), the tool annotations for each software environment themselves (in .ttl format) and the task specifications in data type formats are fed into APE which subsequently generates a set of up to *n* distinct workflow solutions of length *k* for each goal specification (Kruiger et al., 2020). For this research, the synthesis engine will require as configuration the annotated databases from software environments, the generated GIS taxonomy as well as the task specifications from the geo-analytical questions to synthesise the workflows.

3.3 Workflow Translation

The ability to automatically translate GIS workflows would support the sharing and re-use of workflows between GIS analysts and software environments easier where extensive knowledge of both environments is not present. Additionally, the methodologies and semantic technologies developed for the purpose of automatic workflow translation could also support the generation of a tool repository covering tools from various GIS software environments. Such a repository would be useful in several ways. Firstly, such a tool repository would allow analysts to query for tools based on a specific set of semantics or functionality with the purpose of comparing tools across a range of software environments. This querying supports the discovery of tools in various environments without extensive knowledge of the environment itself and allows for greater freedom in software usage based on a given set of requirements. This supports the potential for more relevant and comprehensive GIS analysis through efficient workflow synthesis and/or translation. Whilst the development of this repository, the challenges and opportunities related to this, is beyond the scope of this research, the semantic technologies developed for the purpose of workflow translation provides a preliminary step to its development based on the abstraction of tools beyond software environments through the development of annotated databases and taxonomies.

The need for storing and creating workflow repositories to support workflow storage, retrieval and reuse of workflows with the goal of improving the accessibility for non-specialists has been discussed extensively in the literature (Gil, 2007; Deelman et al., 2009; Bergmann & Gil, 2014; Ubels, 2018). The efficient retrieval of the workflows in this repository is often supported by semantic technologies wherein semantic workflow representations enrich the workflows in these repositories by adding metadata and constraints that allow for improved findability and searchability (Bergmann & Gil, 2014). This literature, however, does not explicitly discuss the translation of these workflows where their reuse requires a translation between software environments to perform an analysis across software environments. Additionally, there is very little literature which discusses the importance and practicalities of developing tool repositories using semantic technologies for non-software-specific workflow translation or generation to improve the efficiency and comprehensiveness of GIS analysis. What is available on workflow translation only discusses the translation of workflow languages for improved readability (van der Aalst & Lassen, 2008) or the translation of workflows between visual and script-based representations using semantic technologies (Maheshwari & Montagnat, 2010). While these provide practical insights into the use of semantics for equivalence in translation, they do not provide a comprehensive framework for translation workflows between GIS software environments.

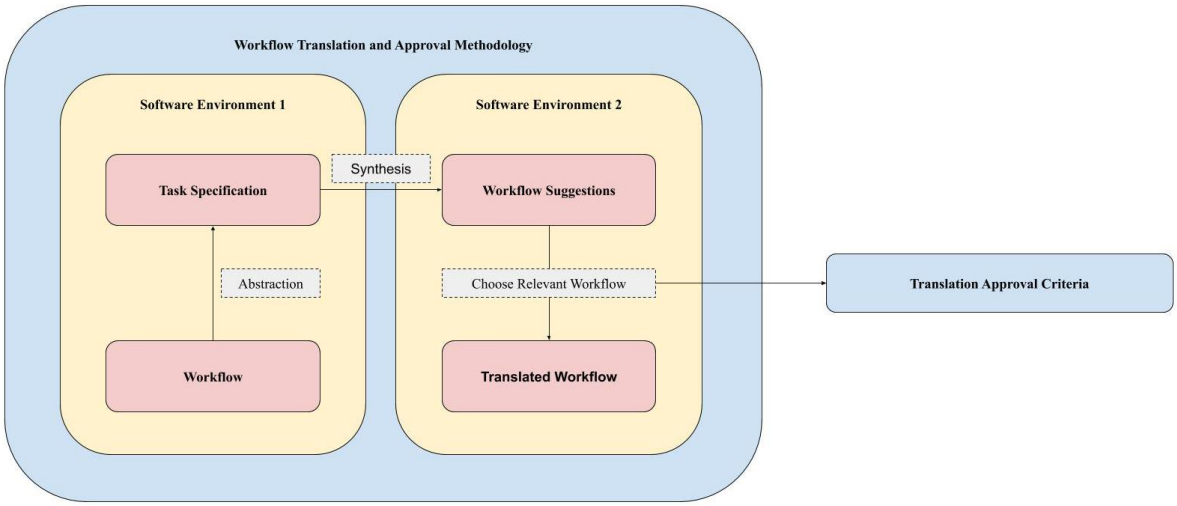
It is within this gap in the literature that this research thesis situates itself and will build upon what literature there is available on both workflow abstraction, workflow synthesis and (where available) workflow translation. Indeed, this research thesis aims to provide a methodology for the automatic translation of GIS workflows between two software environments using semantic technologies; the practical formalisation and implementation of which will be discussed in the Methodology chapter (Chapter 4) of this thesis. The semantic technologies specifically required for the translation of GIS workflows will be discussed from a theoretical perspective here, building on the research on GIS workflow synthesis outlined in the previous section.

3.3.1 Loose programming for GIS Workflow Translation

Although not explicitly explored in the literature to date, loose programming, on which automatic workflow synthesis mechanisms are based, may also be useful for supporting the automatic translation of GIS workflows in several ways. Firstly, the automatic synthesis of workflows is achieved through the underspecification of workflows in both the horizontal and vertical dimensions (Lamprecht et al., 2010). Because the way in which these processes and connections are implemented in a workflow is often software-specific, the process of underspecification removes the requirement that software-specific details of a workflow are needed for an automatic workflow synthesis or, indeed, translation. Secondly, because the process states of a workflow are underspecified in this way, the synthesis mechanism relies on the definition of workflow task specifications, which include the semantic properties of input data and the goal state of the workflow, for the definition of workflow tool inclusions, constraints and meaningful analysis outcomes. Recent work has used data type signatures specified by the CCDT ontology to semantically defined these task specifications, data types that are purposefully developed as non-specific to any software environment (Meerlo, 2019; Kruiger et al., 2020). To put this more concretely, workflow goals are only defined based on the types of their data inputs and outputs and not based on software-specific constraints such as operations and processes. The abstraction of the workflows in this way allows for workflow to be essentially synthesised and translated for use in any GIS software environment.

In making use of data type signatures, the CCDT ontology is also well placed to ascribe semantic meaning to tools beyond the software environments in which they originate for the purpose of building annotated tool databases. These annotated databases can be formalised for any GIS software environment and subsequently used to generated workflows using a configured synthesis mechanism and task specification. By extension then, the translation of workflows between software environments is, therefore, essentially achieved through the abstraction of tools based on their annotation using a non-software-specific ontology, the definition of workflow input and out goals using said ontology as well as the use of a synthesis mechanism making use of underspecification. The process for this proposed translation mechanism based on loose programming is illustrated in the figure below (Figure 3.8).

Figure 3.8. Proposed Automatic Translation Methodology based on Loose Programming



Based on this proposed mechanism, there are several required technical elements for workflow translation which will be theoretically explored in the following subsections of this chapter.

3.3.2 GIS Workflow Translation Ontologies

The formalisation of an ontology necessarily requires that one is able to capture a (subset of a) particular domain of knowledge (Zuniga, 2001). In a similar manner to an ontology used for GIS workflow synthesis, an ontology used for GIS workflow translation would need to find a method of formalising knowledge related to GIS workflow processes and operations as well as GIS tools and their data inputs and outputs in a manner which is relevant to all software environments. What semantics surrounding this knowledge are meaningful to formalise for workflow translation specifically with such an ontology also needs to be considered (Scheider et al., 2016). Such an ontology can then be used to develop annotated tool databases for various sets of GIS tools across numerous software environments in line with those created for the purpose of workflow synthesis (Meerlo, 2019; Scheider et al., 2020). These databases would develop a machine-readable tool repositories of GIS tools from different environments share common descriptions based on, for example, their data inputs and outputs. The ontology which is currently best suited as well as descriptive and extensive enough to develop an annotated tool database for this purpose is Scheider et al.'s (2020) CCDT ontology. This ontology is also most suitable because it has already been used and evaluated for workflow synthesis, the methods of which are closely aligned with the translation methodology used in this thesis. The CCDT ontology supports the description of GIS tools based on their data input and output types. Because having data input and output types is a common feature for all GIS tools irrespective of software environments, annotating tools based on this ontology allow for the meaningful annotation of tools as well as creating a certain level of abstraction of said tools from the software environment from which they originate. This allows for the development of semantically comparable databases to be created across a range of GIS software environments. The practical development method for creating these annotated databases is outlined in the Methodology Chapter (Chapter 4) of this research thesis.

3.3.3 APE Configuration for Workflow Translation

Because the role of APE within the workflow translation mechanism is essentially the same as its role within workflow synthesis, the configuration of the synthesis engine requires the same inputs. Firstly, the configuration of APE requires as input a GIS taxonomy based on the CCDT ontology and an annotated tool database (Kasalica & Lamprecht, 2020a). Because the GIS taxonomy is based on the annotated tool databases, and because a translation mechanism for the purpose of this research thesis requires two tool databases, two GIS taxonomies making use of each annotated tool database needs to be created; a requirement specific to the translation of workflows using the proposed mechanism. When configuring APE, therefore, the GIS taxonomy corresponding to the relevant annotated tool database needs to be used as input. Secondly, the configuration of APE for workflow translation requires the input of annotated tool databases. When configuring the engine for workflow translation, the annotated tool database used as input needs to match the goal software environment for the workflow. Lastly, APE also requires as configuration input the abstraction of a specific workflow from a specific software environment into semantically described workflow inputs and outputs based on the relevant competency question. In line with those used for synthesis, these are presented in the form of task specifications where input and goal specifications are defined using their data type semantics.

3.4 Workflow Quality Evaluation

The evaluation of GIS workflows as part of this research thesis is required to assess whether the CCDT ontology and the translation mechanism can be used to generate quality workflows that produce meaningful outputs from a range of software environments (Scheider et al., 2016). Whilst the need to evaluate workflows is not without precedent, the body of literature available on GIS workflow quality evaluation is limited. Indeed, the literature which does discuss evaluation techniques almost always present workflows as Petri nets where workflow processes and outputs come at a cost to workflow inputs or resources (van der Aalst, 1998; Clempner, 2017). This conceptualisation for workflow evaluation does not accurately model GIS workflows because the data in a GIS workflow continues to exist no matter how many times processes are carried out with the data as input. There is currently only one formalised method of evaluating GIS workflow quality. This method is defined by Kruiger et al. (2020) for the evaluation of workflows generated by an automatic synthesis mechanism, the mechanism which this research builds on in attempting to translate workflows between environments. Because this synthesis mechanism forms the basis of the translation mechanism used in this research, an adapted version of this framework may be useful in the context of workflow translation, where the automatic translation mechanism and underlying ontology will be assessed in a step-wise method. This adapted framework will be presented in the Methodology chapter of this thesis following the presentation of the supporting literature in the subsections that follow.

3.4.1 Individual Workflow Quality Evaluation

Workflows can, in general, be evaluated for quality either at the time of design (design-time) or at the time of their implementation (run-time) (Kruiger et al., 2020). While a run-time evaluation of the workflow can highlight whether the workflow is actually executable, the fact that it is executable does not necessarily mean that the workflow is of a high quality due to the fact that an executable workflow may still produce a meaningless output (Scheider et al., 2016). As such, a design-time evaluation of the workflow quality is argued to be more suitable to assess this meaningfulness and quality of the workflow. A design-time approach to GIS workflow quality evaluation is used by Kruiger et al. (2020) as part of their evaluation of the CCDT ontology used for workflow synthesis, the same ontology used in this research thesis. This approach to workflow quality evaluation is based on approaches used for information retrieval evaluation where ‘workflow synthesis is treated like a retrieval process, and its precision is measured by the extent to which the synthesised workflows [can] answer a given question’ (Kruiger et al., 2020).

In principle, synthesised workflow evaluations based on information retrieval methods could be evaluated based on precision and recall, where the former assesses the proportion of retrieved answers that are correct given all retrieved answers and the latter assesses the proportion of retrieved answers that are correct given all correct answers (Kent et al., 1955; Kruiger et al., 2020). Kruiger et al. note that while a framework based on recall would, of course, provide the most comprehensive method of quality evaluation, this method can be difficult and problematic because it requires that GIS experts be aware of all possible correct workflows for a given competency question (2020). As such, the evaluation framework used by Kruiger et al. (2020) to evaluated synthesised workflows utilises a precision framework with four workflow error types at two different severity levels as outlined below. This evaluation framework can be utilised by an expert to manually evaluate the quality of synthesised workflows based on the following error types. The error types used for workflow quality evaluation are summarised in the following table (Table 3.5).

Table 3.5. A summary of the different error types possible in synthesised workflows.

Error Severity	Error Type	Example Workflows
Hard	Signature	Figure 3.12b
	Semantic Imprecision	Figure 3.12a
Soft	Redundancy	Figure 3.13b
	Data Quality	Figure 3.13c

a. Hard Errors

Hard workflow error types are defined as errors which result in either a completely invalid or meaningless workflow outputs or a workflow that cannot be executed due to the combination the wrong data formats. These hard errors are composed of two subtypes, namely, signature error and semantic imprecision error (Kruiger et al., 2020).

Signature Error

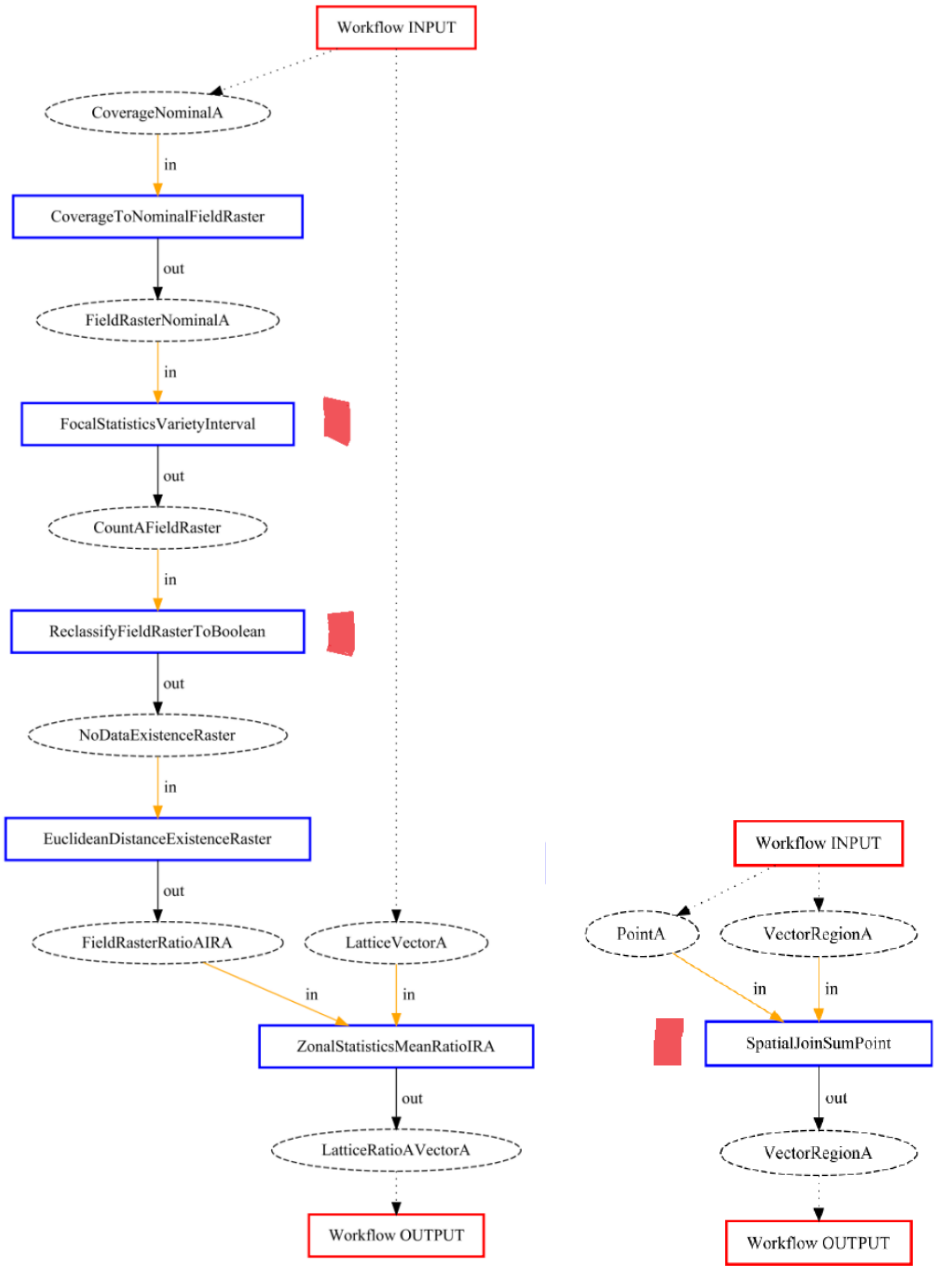
A signature error is defined as an error wherein a workflow or a part thereof cannot be executed because a tool has been incorrectly applied to the workflow. Typically, this error occurs as a result of two issues. Firstly, this error can be produced where the synthesis engine has placed two incompatible tools together due to a misinterpretation of data input types. Although correct interpretation of input data types to produce valid tool combinations is what makes a synthesis engine useful, this error is still likely where a synthesis engine is still under development as is the case with the APE synthesis engine, for example. Secondly, the signature error can be due to human error in a tool annotation. Where the latter is the case, this highlights a relatively easy fix in the tool annotation, and as such a signature error based on human error is unlikely to be reported. Any workflow which includes a tool which should not be valid given valid semantic constraints is categorised as a signature error according to the framework.

Semantic Imprecision Error

The CCDT ontology provided by Scheider et al. (2020) for the formalisation of geospatial knowledge, as the only comprehensive formalisation of geo-analytical knowledge available for the purpose of workflow synthesis, is based on three elements, namely, geometric layer types, spatial core concepts and measurement levels present in spatial information. Whilst this ontology has provided a comprehensive formalisation of geo-analytical knowledge, it is not a complete documentation of knowledge related to geo-analysis and, therefore, may introduce error types where tools and goal specifications cannot be fully annotated with enough semantic detail. As such, semantic imprecision errors tend to produce meaningless or invalid outputs for a given competency question because the ontology is missing some required semantic constraint on data, tools or information which is present in the natural language question. Any workflow that correctly achieves the given goal specification but does not answer the relevant competency question can be considered to contain a semantic imprecision error type. The identification of this error type would beg the conclusion that the ontology used is not extensive enough to capture all nuances included in natural language questions. Naturally, although this error type would reduce the validity of the ontology used for workflow synthesis, it also highlights where the ontology lacks nuance and could be improved.

The following figures (Figure 3.12a and Figure 3.13b) illustrate the two types of hard errors in practice as provided by Kruiger et al. (2020). As explained by the authors, all three workflows either result in a meaningless answer to the question ‘What is the accessibility of parts for each PC4 area in Amsterdam?’ and the question ‘What is the number of sports facilities in each PC4 area?’ or the workflow is simply not executable. In the case of Figure 3.9a, the workflow is generated based on the CCDT ontology to answer the first question where it converts land use polygons to a land use raster and counts the variety of land use types in a neighbourhood around each raster cell. This diversity in land use types in then reclassified to an existence raster with a Boolean measurement level and then Euclidean distance is applied to calculate the distance to a filtered land use type. An average distance to this filtered land use is then calculates for each PC4 area. This output is not valid for the competency question being asked and, therefore, this is classified as a semantic imprecision error (Kruiger et al., 2020). In the case of Figure 3.9b, a workflow is generated to answer the second question using the state-of-the-art (SOTA) ontology. In this workflow, the points of the sport facilities at a nominal measurement level are summed for each PC4 area. Because nominal attributes cannot be summed based on their encoding as strings, this workflow would be inexecutable and is classified as a signature error (Kruiger et al., 2020).

Figure 3.9. Examples of Hard Errors in Workflow Outputs following Workflow Synthesis Erroneous function applications are highlighted with red stripes (Kruiger et al., 2020).



a. Semantic imprecision error in workflow synthesised for the question 'What is the accessibility of parks in each PC4 area in Amsterdam?'

b. Signature error in workflow for the question 'What is the number of sports facilities in each PC4 area?'. It should be noted that this workflow in synthesis based on a state-of-the-art (SOTA) ontology used in Kruiger et al.'s (2020) research but the underlying principle of the error is the same for the purpose of this research.

b. Soft Errors

The identification of a soft error renders the workflow of a lesser quality than the gold standard workflow but does not affect the workflows ability to sufficiently answer the relevant competency question. There are two sub-types of soft errors.

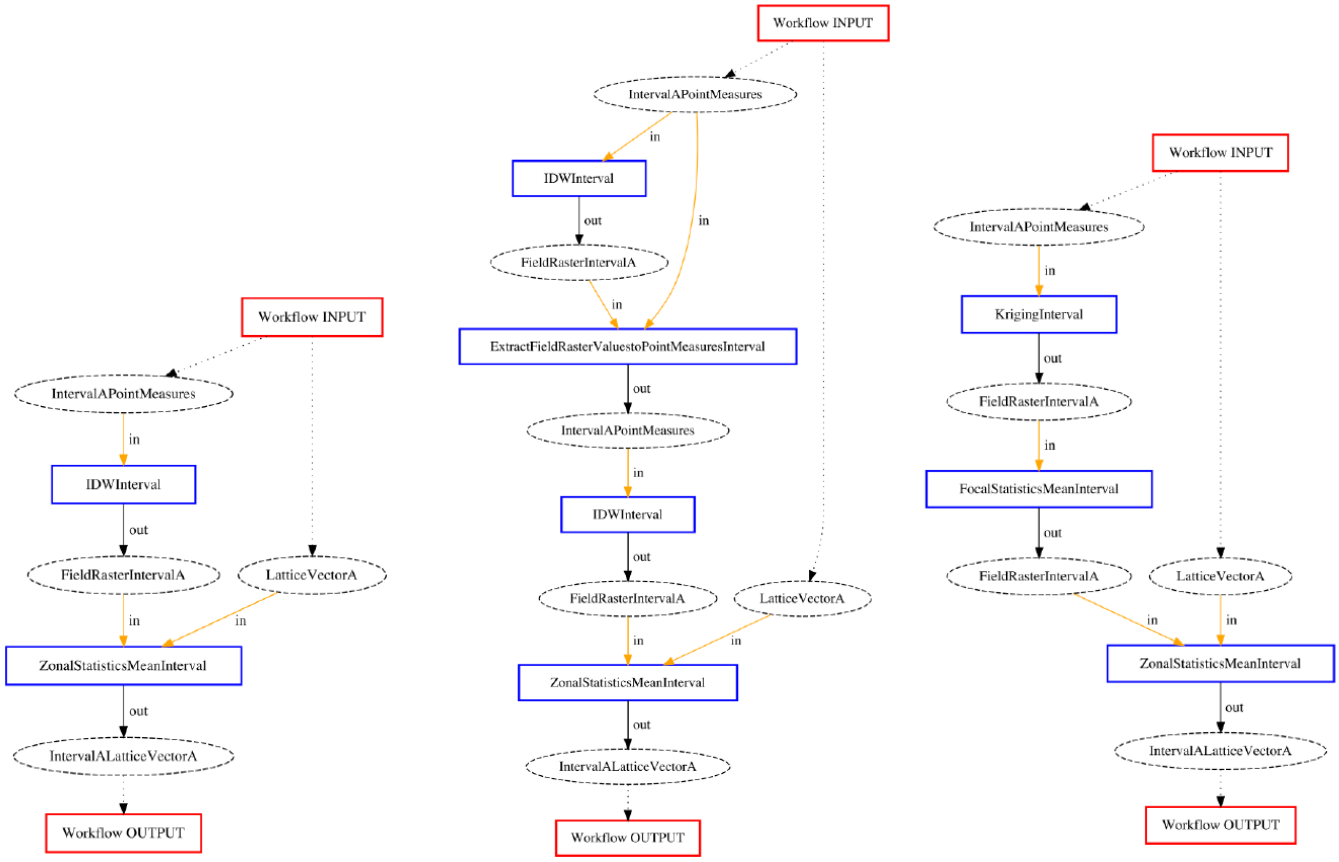
Redundancy Error

A redundancy error is classified as a redundancy error where workflows are making use of tool applications which are not essential in producing a valid answer to the relevant competency question. An error is classified as a redundancy error for two reasons. Firstly, tool redundancy can happen in a workflow where the tool in question is not connected in terms of output to the final workflow output. While this does not necessarily affect the quality of the final output or workflow, it clutters the workflow unnecessarily. This redundancy is a result of the way in which the workflow synthesis engine is set up in that the engine seeks to identify all possible workflows that fit the goal specification regardless of whether those tools are useful. Secondly, tool redundancy can also happen where a completely valid workflow is re-synthesised with a slight adjustment of the order of tools or movement of a process down the temporal timeline. A readjustment of tools, as long as the tools are used before the necessary operation requires them as input, still allows for valid workflows but these are not fundamentally different from the originally valid workflow (Son, Sun Kim & Ho Kim, 2005).

Data Quality Error

An error is considered a data quality error where transformations in data have resulted in a reduction in geodata quality unnecessarily. Geodata quality is measured along a few dimensions, including but not limited to, positional and attribute accuracy, granularity or resolution and completeness (Guptil and Morrison, 2013 as cited by Kruiger et al., 2020). The quality errors identified in this category generally result from an unnecessary reduction in spatial resolution. This is common where, for example, a raster has been transformed into a vector because, while valid, this type of transformation will change the resolution of the data and, therefore, reduce its quality. Another way in which data quality can be affected during workflow synthesis is where attributes have been reclassified with different levels of measurement. It is common practice to always use the highest common measurement level when reclassifying data to ensure the highest level of information. However, in a similar manner to tool redundancy, the synthesis engine may synthesise multiple workflows with different measurement levels and, therefore, reduce the quality of the workflows where this is the case. The following figure illustrates both types of soft errors present in a synthesised workflow. These workflows were synthesised to answer the question ‘What is the average temperature within each PC4 area in Amsterdam?’ using the CCDT ontology.

Figure 3.10. Examples of different soft error types for workflows synthesized for the question 'What is the average temperature within each PC4 area in Amsterdam?' using the CCDT ontology (Kruiger et al., 2020).



a. Correct, no soft errors b. Correct but redundancy error c. Correct, but data quality error

While workflow quality evaluation based on error types allow for the assessment of workflow quality based on a workflow by workflow basis, additional steps need to be taken in order to assess the capacity of the synthesis mechanism to produce quality individual workflows as well as to make conclusions on the capacity of the ontology to support the synthesis of quality workflows.

3.4.2 Synthesis Engine/Translation Mechanism Evaluation

Perhaps as a result of the general lack of literature on GIS workflow evaluation or due to the novelty of workflow synthesis in the field of GIS, there also exists a general absence of literature on the evaluation of the capacity of a synthesis mechanism to produce quality workflows. Because part of the translation mechanism is essentially a workflow synthesis mechanism, assessing this translation mechanism in a similar way for its ability to reproduce quality workflows is essential to this research. Meerlo (2019), in research on the implementation of an automatic workflow synthesis mechanism has suggested that in order to assess the capability of the synthesis mechanism to produce quality workflows the generation of workflows by synthesis engine be approached as a classification problem; a common form of information retrieval quality measurement. A classification problem in this context would mean that generated workflows be placed into categories based on a presence of some value to draw some conclusion from the

values observed. Because the evaluation of a synthesis mechanism is primarily assessing the capability of the mechanism to produce quality workflows, the error types presented in the previous subsection, in assessing the quality of a given workflow based on the presence of error, is a useful categorisation method for the purpose of synthesis mechanism evaluation. A common approach to evaluating the performance of a classification problem, in this case the generation of workflows by a synthesis mechanism, is a confusion matrix (Table 3.8). The confusion matrix differentiates classifications along two dimensions, namely, actual and predicted, and as either a positive or a negative. As such, classifications using this matrix will fall into four categories (Stehman, 1997).

Table 3.6. Confusion Matrix

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

1. **True positive:** A true positive is recorded if a workflow is deemed valid by the GIS expert and is predicted as such by the workflow synthesis mechanism.
2. **False negative:** A false negative is recorded if a workflow is deemed valid according to the GIS expert, but the synthesis mechanism determined the workflow to be invalid.
3. **False positive:** A false positive is recorded if the workflow is deemed invalid by the GIS expert but is predicted as a valid workflow by the synthesis mechanism.
4. **True negative:** A true negative is recorded if the workflow is deemed invalid by the GIS expert and is predicted as such by the synthesis mechanism.

A confusion matrix allows for the measurement of four metrics of classification, namely, accuracy, precision, recall, and specificity based on the combination of the four categories outlined above. In practice, the synthesis mechanism using the synthesis engine will only produce workflows it deems valid based on the achievement of the goal specification supplied in the task specifications. As such, false negative and true negative identifications within the set of workflows generated is not possible because no invalid workflows as deemed as such by the synthesis engine will be presented in the set. However, a false negative can be identified where the GIS expert-produced workflow is not presented in the set of workflows generated by the workflow synthesis engine. Although this would make the assumption that there is only one possible workflow for a given competency question, which is unlikely to be true, measuring recall in this context may at least indicate the ability of the engine to make correct workflow suggestions. This will be discussed further in the results chapter of this thesis. The presence of hard and soft error types in the workflows as manually identified by the GIS expert will place the workflows into either the false positive or true positive category, respectively. As a result, both the precision and the recall capabilities of the synthesis mechanism can be calculated to some degree.

Because the nature of the synthesis mechanism means that a true negative classification is not possible and a false negative classification only possible in the absence of the expert-predicted correct workflow in the

workflow suggestion set, only precision and recall can be measured using the confusion matrix. Precision can be calculated by dividing the number of correct workflows as predicted by both the engine and the expert (true positives) by the total number of workflows that were predicted to be correct by both the expert and the engine (true positives) as well as the only the engine (false positives) (see Equation 1).

Equation 1. Precision Equation

$$precision = \frac{no.of\ true\ positives}{no.of\ true\ positives + no.of\ false\ positives}$$

The result of the precision calculation denotes what proportion of workflows that were predicted to be correct were actually correct and as such, the higher the precision method, the better the capability of the synthesis mechanism to generate high quality suggestions which are able to meaningful answer the competency question posed.

Recall can be calculated by dividing the number of correct workflows as predicted by both the engine and the expert (true positives) by the total number of workflows that were predicted to be correct by both the expert and the engine (true positives) as well as only the expert (false negatives) (see Equation 2).

Equation 2. Recall Equation

$$recall = \frac{no.of\ true\ positives}{no.of\ true\ positives + no.of\ false\ negatives}$$

The result of the recall calculation expresses the ability of the synthesis engine to find all correct workflows for a given competency question. As such, the higher the recall, the greater the capability of the synthesis engine to make relevant workflow suggestions.

3.4.3 Ontology Evaluation

In a general sense, it is necessary to evaluate an ontology with the goal of developing these ontologies for use and reuse in a range of (broader) contexts. Indeed, within the context of GIS workflow synthesis, for example, the CCDT ontology as provided can potentially be used in both workflow synthesis and workflow translation between software environments and, therefore, its performance in both these areas needs to be assessed. It is from this perspective that ideas and preliminary guidelines are discussed in the literature around appropriate evaluation criteria and any errors present in these (Gomez-Perez, 2001; Gangemi et al., 2005; Raad & Cruz, 2015). Despite this, no unified approach to ontology evaluation is currently available. Gangemi et al. (2005) do, however, provide the most extensive and easily implementable categorisation of methods for ontology evaluation. Here, the authors discuss three categories, namely, structural measures, functional measures, and usability-profiling.

Structural measures

This category of ontology evaluation focuses on syntax and formal semantics included in the ontology. Here, several structural measures are defined, namely, depth, breadth, tangledness, leaf and sibling distribution, density, modularity, consistency etc. Each of these measures is able to 'identify topological, logical and meta-logical properties by context-free metrics' in evaluating the ontology itself.

Functional measures

This category of ontology evaluation is focused on the intended use of the ontology in question wherein the ontology is evaluated for its ability to accurately specify the domain knowledge when implemented in a given context. Indeed, this evaluation technique measures the extent to which the ontology is able to mirror a certain field of expertise (Steels, 1990 as cited by Gangemi et al., 2005). In the case of GIS workflow synthesis, for which this evaluation is perhaps most appropriate, this evaluation method would evaluate the quality of the output of the workflow and then infer the functional capacity of the ontology through output matching.

Usability-profiling

This evaluation method focuses on the measurement of the ontological profile which addresses the communication context surrounding the ontology in question where the ontology profile is defined by the metadata and relevant annotations, such as authorship, interfacing etc. of the ontology in question. There are three distinct levels within this evaluation technique, namely, recognition, efficiency, and interfacing. Concretely, this evaluation method assesses whether an ontology is easily used based on the availability and completeness of its metadata.

Based on these categories, the use of functional measures to assess the performance of the ontology is perhaps the most appropriate method in the context of GIS workflow synthesis. Indeed, the framework presented by Kruiger et al. (2020) for evaluating the CCDT ontology against the SOTA ontology makes use of the functional measures of the ontology. Here, a synthesis engine is calibrated to generate a set number of workflows with a maximum length or number of tools possible in the workflow. This calibration makes it possible to produce a set of workflows for a given competency question that can subsequently be reviewed for workflow errors and, where necessary, classify these errors according to error type as presented in sub-section 3.4.1. To evaluate the capacity of an ontology to support quality workflow synthesis, the presence of these errors within a set of workflows are counted and categorised according to their error type for each competency question (Kruiger et al., 2020). A precision calculation can then be implemented wherein the number of correct workflows is divided by the whole set of workflows, both correct and erroneous, generated for each question to produce a precision statistic. It is based on this statistic that conclusions can be drawn as to the capacity of the ontology to support quality workflow synthesis. This calculation can be done per error type to provide some nuance as to which errors are a direct result of imprecision within the ontology and which may have been generated as a result of development issues with the synthesis engine or redundancy in tool inclusions.

3.5 Workflow Similarity Assessment

The use of workflow similarity assessments is not without precedent in the literature. Within the context of scientific workflows, literature discussions generally focus on the need for comprehensive and somewhat automated similarity assessments to improve the retrieval of relevant workflows for a specific task from often large workflow repositories where increased sharing reuse of existing workflows is the end goal (Starlinger, 2015; Koohi-Var & Zahedi, 2018). The emphasis and common understanding of the importance of sharing and reuse has resulted in large scientific workflow repositories (see Taverna-Galaxy[2] and

[2] http://www.taverna.org.uk/documentation/taverna-galaxy/

Kepler[3]) where new challenges are faced with regards to management of these repositories, making these a useful case study for effective retrieval based on similarity assessment.

It is out of this context that the need for comprehensive similarity assessment frameworks is indicated in the literature. Although there exists some debate in the literature as to which similarity algorithms are most efficient (Starlinger et al., 2014), workflow similarity assessments are argued to have the potential for supporting the detection of functionally equivalent workflows, the grouping of workflows in each repository into functional clusters, the use of existing workflows in the development of new ones for application in another scientific field as well as workflow retrieval and discovery based on similarity (Goderis & Goble, 2006; Santos et al., 2008; Silva et al., 2011; Bergmann & Gil, 2012). As such, there are a few approaches to similarity assessment of scientific workflows, the review of which may support the sharing, re-use and, indeed, translation of GIS workflows. These approaches are generally focused around whether an assessment is based on the functional similarity between workflows or the structural similarity of workflows.

Scientific workflows are usually modelled according to their data flows over a pipeline with global inputs and outputs; a structure which typically resembles directed acyclic graphs (DAGs) (Bergmann & Gil, 2012; Starlinger et al., 2015). These workflows when in a repository may be annotated with information such as its title, its keyword tags, the author and/or description of its function, to name a few possibilities. These pieces of information, along with its graph structure, data types and flows, can be used in similarity assessment approaches. Similarity assessment approaches can be grouped according to which pieces of information about a workflow are utilised in carrying out the assessment and, as such, there are generally two categories of workflow similarity assessments, namely, structure-based approaches, and annotation-based approaches (Starlinger, 2015). In the literature reviewed below, there exists some debate as to what the best method for similarity assessment and respective algorithm is (Starlinger et al., 2014b).

3.5.1 Structure-based Similarity Assessments

The nature of SWFM systems, in, for example, attempting to reduce complexity through the limited use of coding, means that workflows developed with these systems often do not have any semantic typing of inputs and outputs; based on which functional similarity assessments are often carried out. Because workflows tend to be generally represented as graphs, structural similarity assessments in terms of, for example, data flows, task modules and data types are more appropriate. Indeed, Starlinger et al. argue that structural similarity assessments, when properly configured, may outperform annotation-based similarity assessments even where rich annotations of workflows are available with the only drawback being that structure-based algorithms for similarity assessment often have a longer run-time than annotation-based approaches (2015).

In general, there are been two approaches to structure-based similarity assessment. Firstly, Goderis & Goble (2006), Santos et al. (2008) and Friesen et al. (2010) make use of isomorphism (i.e. individual or subsequent modules present within workflows) or Maximum Common Isomorphic Subgraphs (MCS) in order to consider and compare workflows based on the similarity in their sub-structures. Starlinger notes that, while this approach is not trivial and needs to be a core issue is similarity assessment, no previous publications have fully described the process of sub-graph or workflow module identification and matching (2015).

[3] https://projects.eclipse.org/releases/kepler

Secondly, Xiang et al. (2007) make use of the structure of the workflow as a whole to compute similarity based on their Graph Edit Distance. In discussing differences in these approaches, Starlinger et al. (2015) stress the fact that annotations should not be excluded from aiding in assessing structure-based similarity between workflows when they are available. Because scientific workflows in graphic form generally illustrate the flow and transformation of data between tasks or modules in a workflow, it is the comparison of data types or flows at workflow task level which tends to be central to structure-based similarity assessment (Starlinger et al., 2014b). These are explored in the following sub-section, along with other potentially useful structure-based similarity assessments as presented in literature.

a. Data-type Comparisons with Jaccard Similarity

Jaccard similarity measures the similarity between two sets of information and is often used as an algorithm to compute the similarity between two sets of information. In the context of scientific workflows this measure of similarity would be used to compare the case workflow and the workflows retrieved from the repository based on a query (Starlinger et al., 2015). Jaccard Similarity is defined as follows:

Equation 3. Jaccard Similarity

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Using this method of similarity assessment, workflows can be compared to another workflow to assess similarity based on the presence of overlapping data types in their respective structures or substructures. Starlinger et al. (2014, 2015) make use of the Jaccard similarity assessment as part of their work on structure-based workflow similarity assessment for workflows housed in each repository. Additionally, Neo4j[4] make use of this approach in computing comparisons between nodes present in their graphic databases.

b. Sequence Comparison with Ratcliff-Obershelp Similarity

The Ratcliff-Obershelp similarity measure (Ratcliff & Metzener, 1988) computes the similarity between two strings as the number of characters in two strings which match divided by the total number of characters in both strings combined. This process is done recursively where the characters which were matched in the first iteration are removed from the string and the process is repeated until there are no more matching characters. Although this is most commonly applied to a string or piece of text, this can be applied to workflow semantics such as descriptions, labels or, indeed, data types added to a workflow based on annotations. The use of this method of similarity assessment has not yet been applied to workflow comparison but may have potential within the scope of this research. This will be discussed in Chapter 4.

The following defines the Ratcliff-Obershelp Similarity function:

Equation 4. Ratcliff-Obershelp Similarity

$$D_{ro} = \frac{2K_m}{|S_1| + |S_2|}$$

[4] https://neo4j.com/docs/graph-data-science/current/algorithms/node-similarity/

Where:

- S_1 and S_2 are the strings which are being assessed for similarity.
- K_m is number of matching characters.

c. Workflow Property Comparison with Levenshtein Similarity

Levenshtein Similarity falls into the category of edit distance-based algorithms which compute the number of operations required to transform a string into another (Mayank, 2019). While often used for word-based similarity assessments, Starlinger et al. (2015) point to the possibility of using this method for comparisons of workflows based on distances between module descriptions, labels, and scripts. The transformations which are valid under this similarity assessment include adding or deleting a new character or replacing one character with another. The lower the Levenshtein distance between two strings, the more similar the two strings are to one another. The following defines the Levenshtein edit distance algorithm:

Equation 5. Levenshtein Edit Distance Similarity

$$lev_{a,b}(i,j) = \begin{cases} \max(i,j), & if\ \min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases}, & otherwise. \end{cases}$$

Where:

- $1_{(a_i \neq b_j)}$ is the indicator function equal to 0 when $a_i = b_j$ and is otherwise equal to 1.
- $Lev_{a,b}$ is the distance between the first i characters of a and the first j characters of b.
- i and j are 1-based indices (Levenshtein, 1966).

3.5.2 Hybrid Workflow Similarity Assessment Framework

The Layer Composition (Starlinger et al., 2014, 2015) framework is formed around the argument that the functionality of workflows is determined by the data processing modules it is comprised of and, as such, each module in a workflow needs to be represented as a distinct functional entity to be compared with a module of another workflow in order to draw some conclusion as to whether the workflows are, indeed, comparable as a whole (Starlinger et al., 2014). It should be noted that, although Starlinger et al. (2014) argue that the framework discussed here is a structure-based approach to similarity assessment, the model does also make use of workflow annotations in parts of the framework as will be evident in the following subsections.

a. Pairwise Module Comparison

Starlinger et al. (2014b) argue that in order to apply any structure-based comparison between workflows, workflows should be compared based on their modules or sub-graphs because it is at the level of these modules that data processing occurs in a scientific workflow. It is only after the assessment at this level that the assessment of similarity between whole workflows can be carried out. How modules are compared to one another depends on the type of attributes associated with the module and how uniformly standards of labelling are applied across the workflows in a repository. As such, the framework allows for flexibility in pairwise module comparison by providing the following schemas for its implementation (Table 3.7):

Table 3.7. Pairwise Module Comparison Schema (adapted from Starlinger et al., 2014b).

No.	Schema Name	Schema Description
1	pw0	A default implementation which assigns uniform weights to all module attributes and compares the module type and web-service related attributes using exact string-matching techniques. Module labels, descriptions and scripts are compared using a Levenshtein edit distance measurement.
2	pw3	Compares single attributes in the same way as the previous schema but uses higher, non-uniform weights for labels, scripts, service URI, service name and service authority, in that order. This is a similar approach to Silva et al. (2011 as cited by Starlinger et al., 2014b).
3	pll	Disregards all the attributes but the labels of modules and compares them using Levenshtein edit distance. This is a similar approach to Bergmann and Gil (2012, as cited by Starlinger et al., 2014b).
4	plm	Disregards all attributes but the labels and compares them using script matching techniques. This is a similar approach to Santos et al. (2008 as cited by Starlinger et al., 2014b).

b. Module Mapping

Following the generation of all pairwise module similarities, modules are mapped to each other based on the best similarity match between modules across workflows. When using a singular attribute category to map modules across workflows, such as labels, the mapping is implicitly given as a set of matching modules. If, however, a more complex module comparison procedure based on, for example, a number of attributes has taken place in the previous step, similarity matches between two workflows need to be made explicit before mapping modules to one another. This explicit similarity matching can be done by greedy selection of mapped modules (Silva et al., 2011) or by maximum weight matching (Bergmann & Gil, 2012).

c. Topological Workflow Comparison

With regards to the topological comparison of scientific workflows, the framework makes use of three existing approaches. Firstly, structure agnostic approaches to topological comparison assess similarity based only on whether modules are compared without any overall workflow similarity comparison. Secondly, substructure approaches to topological comparison make use of common subgraphs between two workflows to assess similarity between workflows and lastly, whole structure approaches to topological comparison compare workflows based on their overall structural similarity. The following define these approaches mathematically.

Sets of Modules

Two workflows WF1 and WF2 are defined as a set of modules wherein the additive similarity score of the module pairs mapped by maximum weight matching (mw) is used as the non-normalised workflow similarity. Here, sim(m, m') denotes a module pair's similarity value.

Equation 6. Module Pairwise Similarity

$$nnsim_{MS} = \sum sim(m, m') | (m, m') \in mw(V_{wf1}, V_{wf2})$$

Sets of Paths

This method of topological comparison makes use of the maximum isomorphic subgraph wherein all subgraphs within WF1 and WF2 can be used to compare the workflows based their maximum common subgraph. Here, each workflow is topologically decomposed into a set of paths wherein each path starts from a node without inbound datalinks (source node) and ends without further outbound links (end node). All pairs of matching paths from WF1 and WF2 are compared using the maximum weight non-crossing matching (mwnc) scheme to determine the additive similarity score for the pair of paths. The following defines this calculation.

Equation 7. Additive Similarity Score for Path Pairs

$$sim(P, P') = \sum sim(m, m') | (m, m') \in mwnc(V_{wf1}, V_{wf2})$$

The non-normalised similarity value between WF1 and WF2 is subsequently computed based on a maximum weight matching (mw) of the paths based on their path similarity score previous calculated. The following defines this calculation.

Equation 8. Maximum Non-Normalised Similarity based on Path Pairs

$$nnsim_{PS} \sum sim(P, P') | (P, P') \in mw(PS_{wf1}, PS_{wf2})$$

Graph Edit Distance

The full structures of WF1 and WF2 are compared by computing the graph edit distance using the SUBDUE package. Here, nodes in a graph are labelled and these labels are subsequently used to match similar workflows with each other.

d. Normalisation

Whether or not normalisation is necessary for similarity assessment between two workflows is dependent on the use case. Starlinger et al. (2014b) note that, where workflows of differing lengths are being compared, normalisation should be done to maximise the information about how well two workflows are matching at a global level. Where normalisation is done, Starlinger et al. (2014b) highlight that a slight alteration needs to be made to the Jaccard Similarity index to account for the topological comparisons being made in previous steps. Indeed, the modification made here reflects the fact that sets of modules and paths are mapped to each other based on their similarity, not based on their identity. Where the classic index compares the number of overlapping elements with the overall number of elements, the modification compares the 'amount of similarity of similar elements from the sets with the non-similar remainder' (Starlinger et al., 2014b, pg. 147).

Equation 9. Modification Jaccard Similarity

$$sim_{MS} = \frac{nnsim_{MS}}{|V_{wf1}| + |V_{wf2}| - nnsim_{MS}}$$

e. Annotation-based Measures

Starlinger et al. make use of annotation-based measures which make use of 'textual information recorded with the workflows in a repository' such as labels, free form text descriptions and keyword tags (2015). To assess similarity using these annotations, the authors discuss two approaches to annotation-based similarity methods.

Bag of Words

Workflows are compared by their titles and descriptions using a bag-of-words approach where both are tokenised using whitespace, underscored as separators, converted to lower case and cleansed from any non-alpha numeric characters. The similarity is then computed by dividing total matches by the sum of matches and mismatches. Where #matches are defined as the number of tokens found in both workflow titles or description and #mismatches are defined as number of tokens present in only one of the workflows being assessed (Starlinger et al., 2015).

Bag of Tags

The keyword tags associated with given workflows in a repository are argued to also present an opportunity to assess workflow similarity. The same approach as that used above is applied to this method of assessment wherein #matches are the number of keyword tags that are present in both workflows tags and #mismatches are the number of keyword tags that are only present in one of the workflows being assessed (Starlinger et al., 2015).

3.5.3 Annotation-based Similarity Assessments

Annotation-based approaches to similarity assessment generally make use of the textual or semantic attributes of a workflow, such as metadata relating to descriptions and authors, for example, to cluster similar scientific workflows together. There are several methods for carrying out these annotation-based approaches. Firstly, text mining methods to cluster scientific workflows together based on the textual inclusions in their description is used by Costa et al. (2012) who argue that making use of textual descriptions allows for workflow comparison across different environments or SWFM systems. Indeed, because each SWFM system uses its own syntactic framework for developing and storing workflows, comparisons of semantics or annotations abstract the workflow enough to make a meaningful assessment of similarities with other workflows from the same or different environment contexts; an argument which is interesting in the context of this research thesis. Secondly, Stoyanovich, Taskar and Davidson (2010) have explored the use of tags assigned to workflows in a repository to cluster similar workflows together and conclude that the tag based approach fairs better than a module, structure based approach on large repositories where the range of task applications tend to be much larger. While annotation-based approaches to similarity assessment are generally better for functionality-based similarity assessments, the presence and comprehensiveness of these annotations is a key prerequisite to the success of this approach (Starlinger, 2015). In the case of workflow synthesis and the proposed workflow translation mechanism used as part of this research, where semantic technologies are used to support and implement the synthesis engines, these

annotations are readily available; making workflows generated in this manner potentially well suited to annotation-based similarity assessments approaches.

a. Measures of Semantic Similarity

While workflow similarity assessments based on structural approaches have yielded interesting results, the use of annotation-based approaches may generally be more relevant to this research project due to the prevalence of semantic annotations of the workflows based on their data type flows, workflow modules in the form of GIS tools as well as the natural language descriptions of these tools included in their annotations. As such, the focus of the subsequent review of literature will be focused on measures available to assess semantic similarity in general, rather than only measures available for assessing semantic similarity of workflows, in the interest of providing a broader overview of the methods available. Measuring semantic similarity between concepts is a method of assessing the semantic distance between two concepts in a given ontology or corpus where similar concepts share common characteristics (Slimani, 2013). In general, there are four categories of semantic similarity measures, namely, ontology structure-based measures, information content measures, feature-based measures, and hybrid measures of similarity (Slimani, 2013; Bisht, n.d.) where functions are used to map a pair of descriptions to a real number, usually between 0 and 1, relating to how similar these descriptions are to each other. Because these categories of semantic similarity measures were initially applied to natural language processing or information retrieval contexts, some of measures depend on the presence of a corpus in order to calculate the similarity between concepts found in texts (Resnik, 1999; Slimani, 2013; Bisht, n.d.). This is particularly true of the latter three categories. In the context of workflow similarity measurement, where a corpus is not available or appropriate, these three similarity measurement categories are not applicable. As such, relevant ontology structure-based approaches are described in this sub-section.

Ontology Structure-based Similarity Measures

Structure-based, or edge counting, measures of semantic similarity make use of a given function which is developed to compute the semantic similarity between two concepts based on their position in an overarching ontology hierarchy structure or taxonomy. Here, the concepts are linked together or related to each other based on is-a or part-of relations between concepts and the distance between two concepts is counted based on the number of links between them (Slimani, 2013). The links between nodes or concepts in this measurement approach are not weighted and are only used as single unit distance measures between concepts. Within this category, there are a few different approaches used to calculate semantic similarity based on ontology structure; the most relevant of which in the context of this thesis are presented below. The typology of the ontology structure-based measures presented here are highlighted in the following table (Table 3.8).

Table 3.8. Structure-based Semantic Measures Typology (adapted from Slimani, 2013).

	Data sources	**Semantics**	**Factors**		
			Shortest Path	Concept Density	Least Common Subsumer (LCS)
Path Length	Ontology	Distance	√		
Wu Palmer	Ontology	Similarity	√	√	√

Leacock & Chodrow	Ontology	Similarity	√

Path Length

Path length is a score which denotes the count of edges between two words or concepts in a hierarchy where the shorter the path between the two, the more similar the concepts in question are to one another. The equation used to compute this score is as follows:

Equation 10. Path Length Semantic Similarity Measure

$$Sim_{path}(c1, c2) = number\ of\ edges\ in\ shortest\ path$$

Wu Palmer

This measure considers the position of the concepts (c1 and c2) relative to the position of the Least Common Subsumer (LCS), C. Here, C is defined as a common parent concept to both c1 and c2 that is the closest in path distance or, in other words, is related to both c1 and c2 with the least amount of is-a or part-of links (Bisht, n.d.). The similarity between the two concepts is based on a function of path length and depth in path-based measures and is described by the following equation:

Equation 11. Wu Palmer Semantic Similarity Measure

$$Sim_{wup}(c1, c2) = \frac{2 * N}{N1 + N2 + 2 * N}$$

Leacock and Chodrow

This measure of semantic similarity is a score which denotes the number of edges between two concepts with log smoothing. The following equation is used to calculate this score:

Equation 12. Leacock and Chodrow Semantic Similarity Measure

$$Sim_{LC} = -\log\left(\frac{length}{2 * D}\right)$$

Where length is equal to the shortest path between the two concepts based on edge counting and D is equal to the depth of the taxonomy. The division of the length by two times the depth of the taxonomy in this measure serves to normalise the semantic similarity score calculated.

3.5.4 Semantic Graph-Based Framework for Similarity Assessment

Because the above approaches to measuring semantic similarity were not developed for assessing the semantic similarity between workflows in particular, these do not make sense as full solutions to assessing workflow similarity. While semantic similarity assessment of GIS workflows is limited in the literature, Bergmann and Gil (2012) do present semantic similarity assessment for scientific workflows in the context of workflow retrieval from repositories. Scientific workflows, and indeed GIS workflows, have a strong focus on dataflow between workflow components with the control flow of the workflow being restricted to

a simple ordering of the tasks carried out as the workflow progresses (Ludascher et al., 2009 as cited by Bergmann & Gil, 2012). Here, each component, workflow task or, in this case, GIS tool has a data input, some parameters and a data output which is then consumed by the subsequent task or tool. The links in such a workflow simply indicate the output of one task or tool and the input into another task or tool. This results in a graph-like structure which illustrates data- and control flows in a given process. Using this graph structure to produce semantic representations of workflows through annotations of data and tasks based on a formalised ontology allows for the capturing of the semantics of tasks, data items and control flow items present in a workflow. Bergmann and Gil argue that it is these semantic annotations, such as those produced by Scheider et al., (2020) and Kruiger et al. (2020) for workflow synthesis, which can be used as the basis for similarity assessment (2012) citing case based reasoning for information retrieval as evidence of the success of this method.

a. Representation of Semantic Workflows

In their framework, Bergmann and Gil (2012) represent workflows as semantically labelled directed graphs, where each semantic workflow graph (W) is a quadruple $W = (N, E, S, T)$ and the following is true:

1. N represents a set of nodes within a specific range of types.
2. $E \subseteq N \times N$ is a set of edges within a specific range of types.
3. $T : N \cup E \rightarrow \Omega$ associates to each node and each edge a type from Ω.
4. $S: N \cup E \rightarrow \Sigma$ associates to each node and each edge a semantic description from a semantic metadata language Σ.

b. Modelling Workflow Similarity

Because Bergmann and Gil's (2012) graph-based framework is defined in order to improve similarity assessments between workflows retrieved from workflow repositories, the modelling of workflow similarity with this framework entails the availability of a case workflow CS from an existing repository and a query workflow QS which is used to find similar workflows from a repository. Here, both the constituents and the link structure of both workflows are compared to assess similarity. The authors present a similarity model based on 'an enhancement of the well-known local/global approach for structural CBR' where local similarity measures assess the similarity between two node or two edges of the same type and global similarity measures are obtained by aggregating local similarity values for each graph component (Bergmann & Gil, 2012).

Similarity of Semantic Descriptions

Local similarity assessments are based on the semantic descriptions given to nodes and edges based on the ontology used for annotations and applied using the graphic representation of the workflow. It should be noted that the detailed formalisation of this similarity measure depends on the type of ontology being used in the annotation of the workflows and can be modelled using the global/local approach wherein local similarity measures for each property applied to a node or edge is aggregated to a similarity measure for the entire node or edge (Bergmann & Gil, 2012). Here, semantic descriptions are treated in an object-oriented fashion wherein different properties associated with a single node are treated as individual objects assigned similarity values based on a given ontology or hierarchy of classes. In their similarity model, Bergmann and Gil (2012) do not elaborate as to how similarity values were assigned to classes in the task

ontology used in their research, stating only that the values were assigned during similarity modelling (pg. 121).

Similarity of Nodes and Edges

It is based on the similarity of semantic descriptions that similarity of nodes and edges are defined in the graph-based framework. Here, nodes of different types are considered dissimilar and their similarity always zero and nodes of equal type are defined as having the same similarity value as their semantic descriptions. Edge similarity is slightly more sophisticated in nature compared to node similarity as it does not consider only the semantic description of edges but also the nodes that are linked by these edges. Indeed, two data flow edges should only be similar when they link similar data objects to similar task objects in a workflows.

Similarity of Workflows

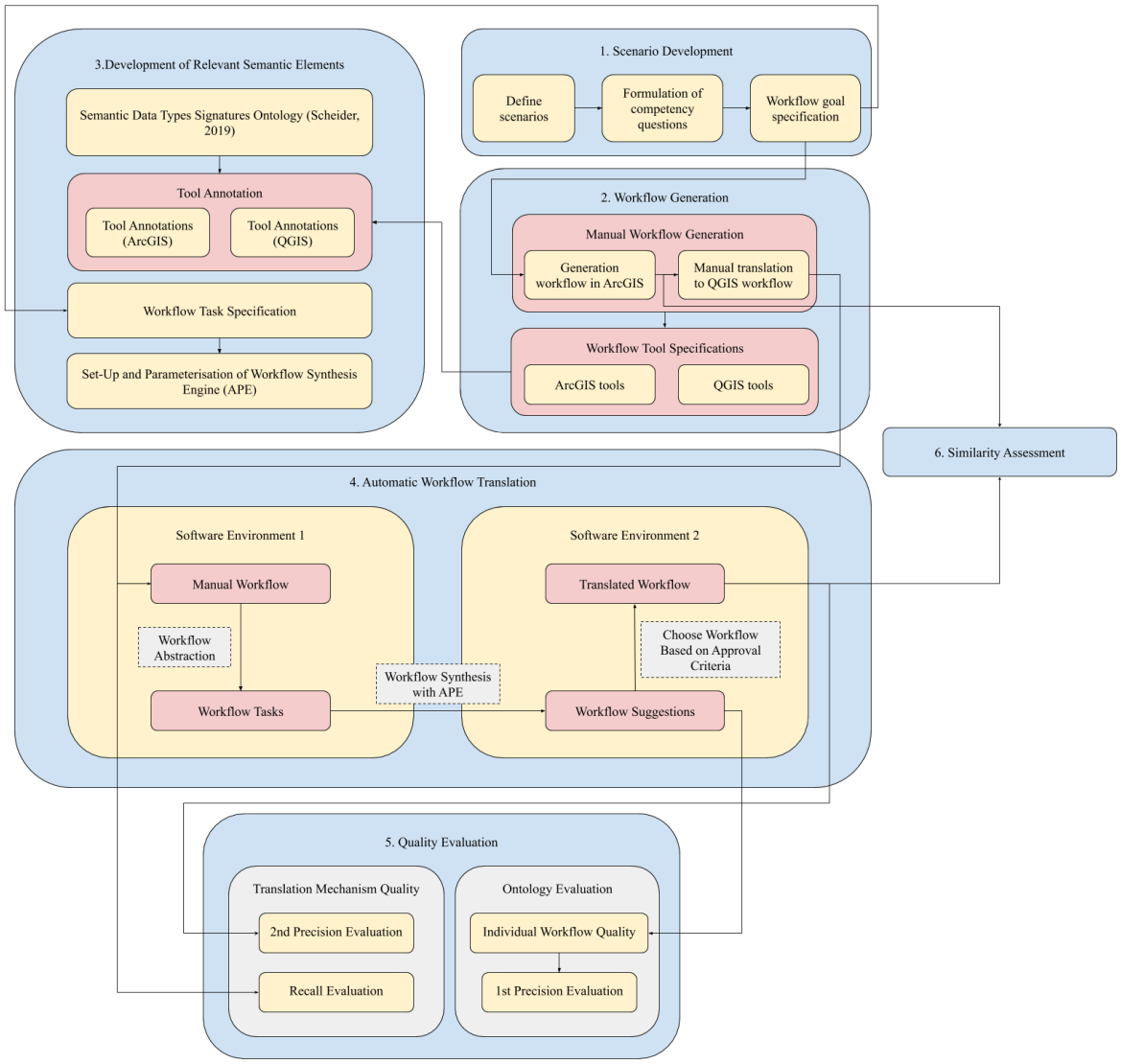
To define the overall similarity between the query workflow QW and the case workflow CW, a legal mapping or type-preserving, partial, injective mapping is initiated by Bergmann and Gil (2012) following a set of constraints. It should be noted that a case node or edge can only be mapped to one query node or edge. It is not necessary to have each case node or edge to be mapped to a query node or edge as the mapping between the two workflows can be partial. Additionally, an edge can only be mapped between workflows if the nodes which connect to that edge are also mapped to the respective nodes in the opposite workflow which are linked by the mapped edge. The similarity between the query workflow and the case workflow $sim_m(QW, CW)$ is calculated based on the aggregation of all the local similarity values of the nodes and edges mapped by m.

In general, the similarity model defined by Bergmann and Gil (2012) can be defined by three model parameters, namely, the similarity function for semantic descriptions, the aggregation function for edges and the aggregation function for workflows. Bergmann and Gil note that each mapping should be interpreted as a particular suggestion for the reuse of the case workflow where the similarity value indicates how useful the mapping is for a reuse opportunity (2012, pg. 122). Because there is a potential for several legal mappings, the overall similarity value should correspond to the highest similarity value of the case to the query workflow.

Chapter 4 Methodology

The purpose of this chapter is to outline the methodology which will be followed as well as the technologies which are to be used in this research. As previously noted, the overarching aim of this research thesis is to highlight the relevance of semantic web technologies for effective GIS workflow translation. To achieve this aim, there are three goals to be met as outlined in the Research Objectives chapter. To meet these goals, this research thesis will be carried out in several methodological stages as specified in Figure 4.1.

Figure 4.1. Methodological Overview



These stages include scenario development, workflow generation, the development of relevant semantic elements, workflow translation and final workflow quality and workflow similarity assessment. Each stage is roughly in line with the research objectives defined in Chapter 2, with the Scenario Development stage

acting as a necessary pre-cursor stage to Research Objective 1. Each stage of the research is divided into sub-stages to support the comprehensiveness of the designed methodology and to highlight the interaction of the research stages with one another. It should be noted that this methodology is not always chronological in nature but is, rather, slightly iterative as clearly illustrated by the flow of sub-stages across research stages.

The first stage of this research thesis, namely Scenario Development, involves the formulation of competency questions based on a predefined, fictional GIS project scenario (Section 4.1). The competency questions used in this research thesis allow for the specification of workflow begin states and goals which are then subsequently used to manually generate the workflows themselves and evaluate the quality of their translation in later research stages. It should be noted that the implementation of the workflows, once generated, are beyond the scope of this research thesis. This research stage is crucial not only for manual workflow generation, tool specification and later automatic translation but also because competency questions serve to aid in the evaluation the translation mechanism and the underlying data types ontology used in this research (Wieleman, 2018). In the case of this research, not only will the capacity of the ontology for workflow translation be evaluated but the translation mechanism will also be evaluated for its ability to produce quality workflows in which the goal specifications in competency questions play a key role. The use of competency questions in this way is not without precedent in the field of GIS (Scheider and Tomko, 2016; Scheider and Ballatore, 2018).

The second stage of this research thesis, namely Workflow Generation, serves to manually generate a set of workflows based on the begin states and goal specifications included in the competency questions. The manual generation of workflows is done in ArcMap using Esri's workflow management system Model Builder. Once generated, these workflows are then used to specify the tools to be annotated using the CCDT Ontology (Scheider et al., 2020) in the second research stage as well as provide the workflows to be manually translated using QGIS's Graphical Modeller workflow management system. As such, this research stage crucially provides the foundations for the technical inputs for both the third and the fourth research stages.

The third stage of this research thesis is the Development of the Relevant Semantic Elements required for workflow translation where the outcome of the research stage will be the configuration of the workflow synthesis engine used to support the translation of workflows. This research stage will take place in three sub-stages. Firstly, the CCDT ontology will be used to annotate the tools specified from manually generated and translated workflows. Secondly, the tasks involved in each workflow will be specified based on their semantic data type properties provided by the CCDT ontology. Lastly, the workflow synthesis engine (APE) will be configured using the relevant GIS taxonomy, the CCDT ontology, the tool annotations for both environments as well as the task specifications; the correct production of which are crucial to the accuracy and quality of the outputs of the translation mechanism. This mechanism will be detailed in the following subsections of this chapter.

The fourth research stage, Workflow Translation, makes use of the semantic elements developed in stage 3 of the methodology to carry out the translation mechanism as a process defined in the previous chapter. Firstly, a given workflow for one software environment is abstracted into workflow inputs and outputs using task specifications for a given competency question. These task specifications are defined based on

their abstract data types and are, in combination with other technical elements, are used to configure the synthesis engine in the previous research stage. The synthesis engine is then run used to generate workflow suggestions for implementation in the second software environment. Which workflows are chosen is based on a set of workflow criteria and is not dependent on the outcome of the quality evaluations and the similarity assessment. This process is repeated for every competency question where ArcGIS workflows are generally translated into QGIS workflows using this mechanism.

The final stage of this research methodology will undertake a quality evaluation in two parts. Firstly, the individual quality of workflows generated by the translation mechanism for each competency question will be evaluated based on error types present in the set of generated workflows. Based on the presence of these error types, workflows will be categorised, and a precision evaluation performed to assess the capacity of the synthesis engine (APE) and the ontology to produce quality workflows. Following this, and based on the error type categorisation, a recall metric will be calculated on the same set of workflows to assess whether the *synthesis engine* is able to final all the correct and relevant workflows for a given competency question. This process is in line with Kruiger et al. (2020). A second precision metric will then be calculated following the final selection of the 'translated workflow' to assess the capacity of the *translation mechanism* to produce meaningful, quality workflows for a given competency question. Finally, a similarity assessment will compute the similarity between the manual ArcGIS workflow and the final translated workflow. This assessment will assist in conclusions about translation quality of workflows and highlight preliminary similarities between the software environments used in this research.

4.1 Scenario Development

The first stage of this research thesis is the formulation of a fictional GIS project scenario on which competency questions can be outlined, analysis goals specified, and input and output states defined. The role of the competency questions developed from this scenario below is twofold. Firstly, the competency questions will provide the basis for generation of the initial workflows in ArcMap's Model Builder. These workflows will, in turn, provide the basis for manual translation of workflows in QGIS. Secondly, the competency questions and their goal specifications will support the evaluation of individual workflow quality, the evaluation of the synthesis engine and translation mechanism as well as the ontology. The scenario used here will serve as the context on which the competency questions will be drafted to generate, test, and evaluate the translation of workflows between software environments in a realistic setting. Each competency question in the scenario will define the question, specify the goal of the workflow in natural language.

4.1.1 Amsterdam Liveability Atlas

The scenario is a fictional GIS project based on an assignment outlined by the University of Amsterdam using openly available data provided by the city of Amsterdam. Here, the task is to create a liveability atlas for neighbourhoods at the level of PC4 postcode areas in the city with a specific focus on the elderly and (some of) there living requirements in the city of Amsterdam (Hwang et al., 2008; Ruth & Franklin, 2014). The competency questions formed based on this scenario have been used for workflow synthesis by Kruiger et al. (2020) and have adapted slightly for this research. This scenario assesses the number of elderly people living in each PC4 area of Amsterdam, the accessibility of green space as well as noise pollution and temperature in each neighbourhood.

Competency Questions

1. What is the number of medical facilities (pharmacies, clinics, hospitals etc.) in each PC4 area?

Given Data	Medical facilities are interpreted as objects and represented by point vectors with a nominal attribute denoting the facility type; PC4 areas in Amsterdam form a Vector Lattice.
Goal Specification	The goal is a Vector lattice on the PC4 level with extensive counts.

2. What is the proportion of elderly people living in each PC4 area in Amsterdam?

Given Data	The CBS Buurt statistics contains percentages of elderly in the population of a neighbourhood, interpreted as a Vector Lattice with intensive, ratio-scaled attributes; PC4 areas in Amsterdam form a Vector Lattice.
Goal Specification	The goal is an intensive, ratio-scaled attribute of a Vector Lattice on the PC4 postcode level.

3. What is the accessibility of parks/green space for each PC4 area in Amsterdam?

Given Data	The CBS land use dataset, called the BBG, can be used to select areas with parks. It is interpreted as a Coverage with nominal attributes denoting the land use type; PC4 areas in Amsterdam form a Vector Lattice.
Goal Specification	The goal is a ratio-scaled attribute of a Vector Lattice on the PC4 level.

4. Which PC4 areas of Amsterdam have noise pollution greater than 70dB?

Given Data	The map of traffic noise levels is interpreted as a Contour map with an ordinal attribute denoting the noise interval in dB; PC4 areas in Amsterdam form a Vector Lattice.
Goal Specification	The goal is an ordinal scaled attribute of a Vector Lattice on the PC4 level.

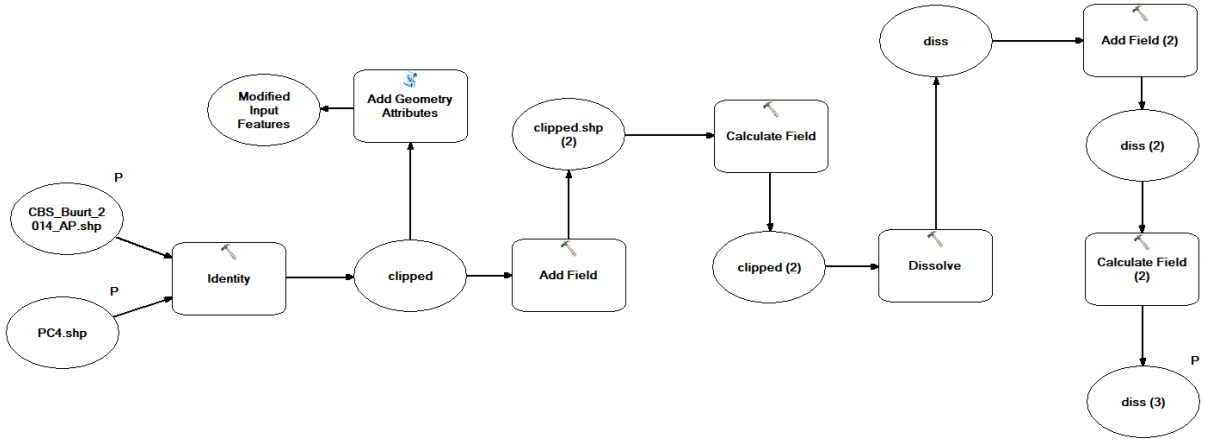
5. What is the average temperature within each PC4 area in Amsterdam?

Given Data	A map of pointwise meteorological measurements with an interval scaled attribute denoting temperature; PC4 areas in Amsterdam form a Vector Lattice
Goal Specification	The goal is an interval scaled attribute of a Vector Lattice on the PC4 level.

4.2 Workflow Generation

Following the definition of competency questions and goal specifications, the second stage of this research thesis is to manually generate the workflows first using Esri's workflow management system Model Builder based on these questions and goals and then subsequently in QGIS's Graphical Modeller. The manually generated workflow for competency question 1 is illustrated in Figure 4.2 (below). These workflows are generated by a GIS expert and represent at least one of the correct workflow solutions to the competency question being posed.

Figure 4.2 Workflow generated in ArcMap for competency question 2: What is the proportion of elderly people living in each PC4 area in Amsterdam?



Following the generation of workflows for all competency questions, each tool in each workflow is specified. Tool specifications involve the definition of tool data inputs and their data types, the tool parameters and operation types where necessary as well as the data outputs. It should be noted that it is common for a tool to have slightly different inputs based on a difference in the parameters and operation rules used. Where this is the case, the tool is annotated as being the product of a number of sub-tools, shown in the table below as 'operation sub-types'. An example of a tool specification is shown in Table 4.1 (below).

Table 4.1 Tool specification for the ArcMap's spatial join tool.

Operation Name	Para-meter	Operation Sub-Type	Input Data Measurement Level	Input Type 1	Input Type 2	Output Data Measurement Level	Output Type
Spatial Join	JOIN_ONE_TO_ONE	Merge rule mean	Ratio	Object Vector	Vector Tessellation	Ratio	Lattice
			Interval			Interval	
			Count			Count	
		Merge rule sum	Ratio	Object Vector	Vector Tessellation	Ratio	Lattice
			Interval			Interval	
			Count			Count	
		Merge rule count	Count	Object Vector	Vector Tessellation	Count	Lattice

Once the ArcMap workflows have been manually translated using QGIS's Graphic Modeller by the GIS expert involved in this research, the tools in these workflows are also specified using the process outlined above. It is these tool specifications that form the basis of the annotated tool databases.

4.3 Development of Relevant Semantic Elements

Following the manual generation and translation of workflows, the third research stage is the Development of Relevant Semantic Elements required for workflow translation; where the outcome of the research stage will be the configuration of the workflow synthesis engine used in the automatic translation of workflows between ArcMap and QGIS. This research phase will take place in three sub-stages, namely: tool annotation and annotated database creation, task specification and the set-up and configuration of the workflow synthesis mechanism.

4.3.1 Annotated Tool Database Creation

The first technical element required for automatic translation of workflows is the development of an annotated GIS tools database. Such a database is created using the CCDT ontology to annotate the GIS tools from both software environments specified in the manually generated ArcGIS and QGIS workflows. The use of this formalised ontology ensures that tools are described in a machine-readable format. Because the tools included in the database are annotated based on the tool specifications, the choice of tools, necessarily, is not a random selection of tools from the tool repositories of the two software environments used in this research. The competency questions formulated for the purpose of this research where purposefully defined to ensure that a wide range of tools were used in the workflows in order provide a diverse range of tools in the annotated database. This is done to be sure that the database is as representative as possible of the range of tools available in each software environment and to maximise the range of tool interactions possible. Additionally, the workflows were first generated using ArcMap software due to the fact that this environment offers comprehensive descriptions and documentation of each tool, its inputs and parameters; thus, supporting the thorough annotation of ArcMap tools. QGIS, as the alternative to ArcMap for the purpose of workflow translation, was chosen because the environment offered a similar set-up for spatial analysis to that of ArcMap. While this environment does not offer as comprehensive a set of tool descriptions, these descriptions are sufficient enough to support tool annotation for the purpose of this research thesis.

The annotation of each instance of tool in the database is a description of valid inputs and respective outputs for the tool in a machine-readable format which, for the purpose of this research thesis, will be done in accordance with the principles of RDF in a turtle format. Turtle serialisation is used over XML serialisation in this research thesis because this serialisation is more easily understood and written in comparison to XML serialisation. RDF principles require annotations to follow a strict structure, an example of which is shown in Figure 4.3 (below). Firstly, each tool annotation starts with the URI of the tool being annotated where the URI denotes the location of the description of the tool on the respective software environment's website; in this case either ArcMap or QGIS documentation. Secondly, a predicate is used to specify either the version of the tool being annotated or the description of the tool using a label predicate. Indeed, the predicate tools:implements is used to specify the version of the tool and rdfs:label is used to add a short description of the tool to the annotation in order to facilitate human understanding of the tool repository. Lastly, the workflow inputs and outputs are specified as the final part of the RDF triple and include a set of

core concept data types to ensure that measurement levels and data input types are semantically comprehensive. Indeed, it is important to note that a tool can have multiple inputs and data type possibilities and this multiplicity in inputs should be annotated accordingly.

Figure 4.3. Annotated GIS tools (ArcMap and QGIS in Turtle format).

ArcMap: Zonal Statistics

```
########## Zonal Statistics
#### Zonal Statistics with Majority Rule
# with Boolean Data
<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/zonal-statistics-as-
table.htm> tools:implements tools:ZonalStatisticsMajorityBoolean.
tools:ZonalStatisticsMajorityBoolean  rdfs:label "Calculates the most common value on values with a Boolean
measurement level of a raster within the zones of another dataset, a zone is defined as all areas in the input that
have the same value. The areas do not have to be contiguous. Both raster and feature datasets can be used for the
zone input.";
   wf:input1 [ a ccd:Lattice, ccd:VectorA ];
   wf:input2 [ a ccd:FieldRaster, ccd:BooleanA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:BooleanA ].

# with Nominal Data
<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/zonal-statistics-as-
table.htm> tools:implements tools:ZonalStatisticsMajorityNominal.
tools:ZonalStatisticsMajorityNominal  rdfs:label "Calculates the most common value on values with a Nominal
measurement level of a raster within the zones of another dataset, a zone is defined as all areas in the input that
have the same value. The areas do not have to be contiguous. Both raster and feature datasets can be used for the
zone input.";
   wf:input1 [ a ccd:Lattice, ccd:VectorA ];
   wf:input2 [ a ccd:FieldRaster, ccd:NominalA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:NominalA ].
```

QGIS: Proximity

```
########## Proximity
#NoDataExistenceRaster input
<https://docs.qgis.org/3.10/en/docs/user_manual/processing_algs/gdal/rasteranalysis.html> tools:implements qtoo
ls:NoDataExistenceRasterProximity.
qtools:NoDataExistenceRasterProximity rdfs:label "Generates a raster proximity map indicating the distance
from the centre of each pixel to the centre of the nearest pixel identified as a target pixel.";
   wf:input1 [ a ccd:NoDataExistenceRaster ];
   wf:output [ a ccd:FieldRaster, ccd:RatioA ].

# FieldRaster input
<https://docs.qgis.org/3.10/en/docs/user_manual/processing_algs/gdal/rasteranalysis.html> tools:implements qtoo
ls:FieldRasterProximity.
qtools:FieldRasterProximity rdfs:label "Generates a raster proximity map indicating the distance from the centre
of each pixel to the centre of the nearest pixel identified as a target pixel.";
   wf:input1 [ a ccd:FieldRaster ];
   wf:output [ a ccd:FieldRaster, ccd:RatioA ].
```

In order to produce a comprehensive annotation of the GIS tools specified, as well as the variety of inputs possible for said tools, the annotation is done by considering how different combinations of semantics describing the inputs, parameters and outputs affect the validity of the tool in practice. For example, if a tool can only accept vector datasets at a count measurement level, then the tool's input is annotated as being

both a vector and count dataset. This, therefore, limits the range of datasets that can be accepted by the tool in a valid workflow and limits the combination of tools that can be put together in a valid workflow by the synthesis engine. Because the measurement level of the dataset significantly limits the inputs accepted by a tool, this needs to be included in the annotation to ensure a tool is valid. Additionally, tools are not only limited in the geometric data types they can accept, but the valid output options of the tool will change depending on the inputs. Indeed, the implementation of a tool accepting a vector dataset at the count measurement level may be able to change the geometry of the vector but is unable to change the measurement level in this case. As such, it is necessary to annotate the tool in such a way that there are sub-tools which accept a subset of the valid input, in this case the count measurement level, and result in a same semantic data type. For the purpose of this research thesis, this consideration of semantic combinations for each tool is done manually.

In developing the GIS tool databases for ArcMap and QGIS, a novel method was performed in annotating tool combinations where the annotation of the individual tools was not logical. This approach essentially annotates combinations of tools to create a *super tool* which is used in given workflow. For example, the ArcMap Select Layer by Attribute tool was combined with the copy features tool to create a *super tool* which performs the select layer by attribute analysis and then copies the selected features to an output dataset. This is used to reduce the potential for redundant complexity and non-sensical tool combinations in the generation of workflows when the synthesis engine is run in the next research phase. An example of the annotation of a super tool is detailed in the following figure (Figure 4.4).

Figure 4.4 Super tool Annotation for Select Layer by Attribute and Copy Features tools

```
########### Supertool: Select Layer by Attribute
tools:FullSelectLayerByAttribute tools:implements tools:SelectLayerByAttributeObjects
   wf:edge _:wf3_1, _:wf3_2;
      wf:source _:in31;
   rdfs:label "a SQL query to select features matching a selection criteria. You build a query expression to
generate the subset. You can query a dataset based on a field in a layer or a table. Using this selection, a new
Feature dataset is created with only the selected features using the Copy Features tool. This feature dataset
represents all the area's that denounce the presence of a certain criteria".
#### Select by attribute is always followed by copy features (treated as one tool in the annotation
_:wf3_1 wf:applicationOf <https://pro.arcgis.com/en/pro-app/tool-reference/data-management/select-layer-by-
attribute.htm> ;
      wf:input1 _:in31;
      wf:output _:out31.
_:wf3_2 wf:applicationOf <https://pro.arcgis.com/en/pro-app/tool-reference/data-management/copy-
features.htm>  ;
   wf:input1 _:out31;
   wf:output _:out32.

#### Object selection variant, needs a nominal attribute
tools:SelectLayerByAttributeRegion
   wf:input1 [ a ccd:ObjectRegion, ccd:NominalA ];
   wf:output [ a ccd:ObjectRegion, ccd:NominalA ];
      tools:algebraexpression    "sigmae objectregions x object y".

tools:SelectLayerByAttributeLattice
   wf:input1 [ a ccd:Lattice, ccd:VectorA, ccd:NominalA ];
   wf:output [ a ccd:Lattice, ccd:VectorA, ccd:NominalA  x object y".
```

```
########## Copy Features
<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/copy-
features.htm> tools:implements tools:CopyFeatures.
tools:CopyFeatures rdfs:label "Copies features from the input feature class or layer to a new feature class.";
   wf:input1 [ a ccd:ObjectVector ];
   wf:output [ a ccd:ObjectVector ].
```

The full list of tools, their super tools and sub tools used for the creation of annotated tool databases are included in the table that follows (Table 4.2). Please note, in the event that the tool had specific object inputs such as point, region or line, these were delineated in the databases but summarised as an ObjectVector for the purpose of the overview table.

Table 4.2 Overview of the Formalised GIS Tools

	Software	Database Format	Operation	Operation Sub-Types	1st Input Type	2nd Input Type	Output Type
Field	ArcGIS	Super tool	Add and Calculate Field		Field Raster, Contour, Coverage		Field Raster
		Super tool	Zonal Statistics and Join Tool	Majority, Mean, Median, Sum	Field Raster	Lattice	Lattice
			Extract Values to Point		Field Raster	Point	Point Measures
			Raster to Coverage		Field Raster		Coverage
		Super tool	Raster to Contour		Field Raster		Contour
			Point Interpolation		Point Measures		Field Raster
			Local Map Algebra		Field Raster	Field Raster	Field Raster
			Clip Raster by Extent		Field Raster	Lattice	Field Raster
	QGIS		Add Field to Attribute Table		Field Raster, Contour, Coverage		Object Vector, Coverage, Contour
		Super tool	Select by Attribute and Extract Selected Features		Coverage		Coverage, Existence Vector
		Super tool	Zonal Statistics and Join Table		Field Raster	Lattice	Lattice
			Zonal Statistics	Mean, Median, Sum	Field Raster	Lattice	Lattice
			r.neighbours	Mean, Median	Field Raster		Field Raster
			Map Algebra		Field Raster	Field Raster	Field Raster

			Clip Raster by Extent		Field Raster	Lattice	Field Raster
		Super tool	Zonal Geometry		Field Raster		Object Vector, Coverage
			Polygonise		Field Raster		Object Vector, Coverage, Contour, Existence Vector
Object	ArcGIS	Super tool	Add Calculate Field		Object Vector, Lattice		Object Vector, Lattice
			Add Geometry Attributes		Object Vector		Object Vector
			Dissolve	Sum, Mean, Count	Lattice		Lattice
			Spatial Join	Mean, Sum	Object Vector	Lattice	Lattice
		Super tool	Select Layer by Attribute		Object Vector		Object Vector
		Super tool	Select Layer by Location		Object Vector	Object Vector	Object Vector
			Euclidean Distance		Existence Raster, Existence Vector, Object Vector		Field Raster
			Feature to Raster		Object Vector		Existence Vector
			Raster to Vector		Existence Raster		Object Vector
			Areal Interpolation		Lattice	Lattice	Lattice
			Clip		Object Vector, Lattice	Object Vector, Lattice	Object Vector, Lattice
	QGIS		Union		Object Vector, Lattice	Object Vector, Lattice	Lattice

			Add Field to Attribute Table		Object Vector, Lattice		Object Vector, Lattice
			Add Geometry Attributes		Object Vector		Object Vector
			Dissolve	Sum, Mean, Count	Lattice		Lattice
			Join Attributes by Location	Mean, Sum, Count	Object Vector	Lattice	Lattice
		Super tool	Select by Attribute and Extract Selected Features		Object Vector		Object Vector
		Super tool	Rasterise Object Dataset and Perform Proximity Analysis		Object Region		Field Raster
			Join Attributes by Field Value		Lattice		Lattice
			Clip		Lattice, Object Vector	Lattice, Object Vector	Lattice, Object Vector

4.3.2 Workflow Task Specification

The abstraction of the workflows required to provide outputs for the geo-analytical questions as defined by the competency questions is achieved through the definition of task specifications. These task specifications are achieved by defining the data inputs and goal specifications for each competency question based on their abstract data types in three dimensions, namely, the relevant core concept type for the data, the relevant geometric type, and the relevant data measurement level. Additionally, as highlighted by the descriptions of the ontology in previous chapters, measurement levels can be subclasses of other measurement levels meaning that when 'NominalA' is used as a measurement level attribute in a tool or goal specification, all other measurement levels are also possible by definition. In order to constrain this where necessary, the use of 'Plain' in, for example, 'PlainIntervalA' constrains the task specifications such that only that specific measurement level is valid as input for the first tool in used in a workflow synthesis. The following task specifications are those used in the configuration of APE based on the defined competency questions.

Task Specifications

1. What is the number of medical facilities (pharmacies, clinics, hospitals etc.) in each PC4 area?

Input Specification	ObjectQ∩PointA∩PlainNominalA; ObjectQ∩VectorTessellationA∩NominalA
Goal Specification	ObjectQ∩VectorTessellationA∩PlainCountA∩ERA

2. What is the proportion of elderly people living in each PC4 area in Amsterdam?

Input Specification	ObjectQ∩VectorTessellationA∩PlainRatioA; ObjectQ∩VectorTessellationA∩NominalA
Goal Specification	ObjectQ∩VectorTessellationA∩PlainRatioA

3. What is the accessibility of parks/green space for each PC4 area in Amsterdam?

Input Specification	ObjectQ∩VectorTessellationA∩NominalA; FieldQ∩TessellationA∩PlainNominalA
Goal Specification	ObjectQ∩VectorTessellationA∩PlainRatioA

4. Where in Amsterdam is the noise pollution equal to or greater than 70dB?

Input Specification	FieldQ∩TessellationA∩PlainOrdinalA; ObjectQ∩VectorTessellationA∩NominalA
Goal Specification	FieldQ∩TessellationA∩PlainBooleanA

5. What is the average temperature within each PC4 area in Amsterdam?

Given Data	FieldQ∩PointMeasures∩PlainIntervalA; ObjectQ∩VectorTessellationA∩NominalA
Goal Specification	FieldQ∩VectorTessellationA∩PlainRatioA

4.3.3 Set-Up and Configuration of Synthesis Engine

There are several steps to the configuration of the synthesis engine (APE) used. The list of inputs required for full configuration and are actively generated as part of this research are included in the table below (Table 4.3).

Table 4.3. Configuration Inputs for APE in the Context of Workflow Translation

Configuration Inputs	Description
Tool Annotations Path	ArcGIS and QGIS Tool Annotations
Ontology Path	GIS Taxonomy
Constraints Path	Competency Question-Specific Constraints
Solution Minimum Length	Minimum length of workflow
Solution Maximum Length	Maximum length of workflows
Number of Generated Graphs	Number of graphs generated by engine
Inputs	Task specifications

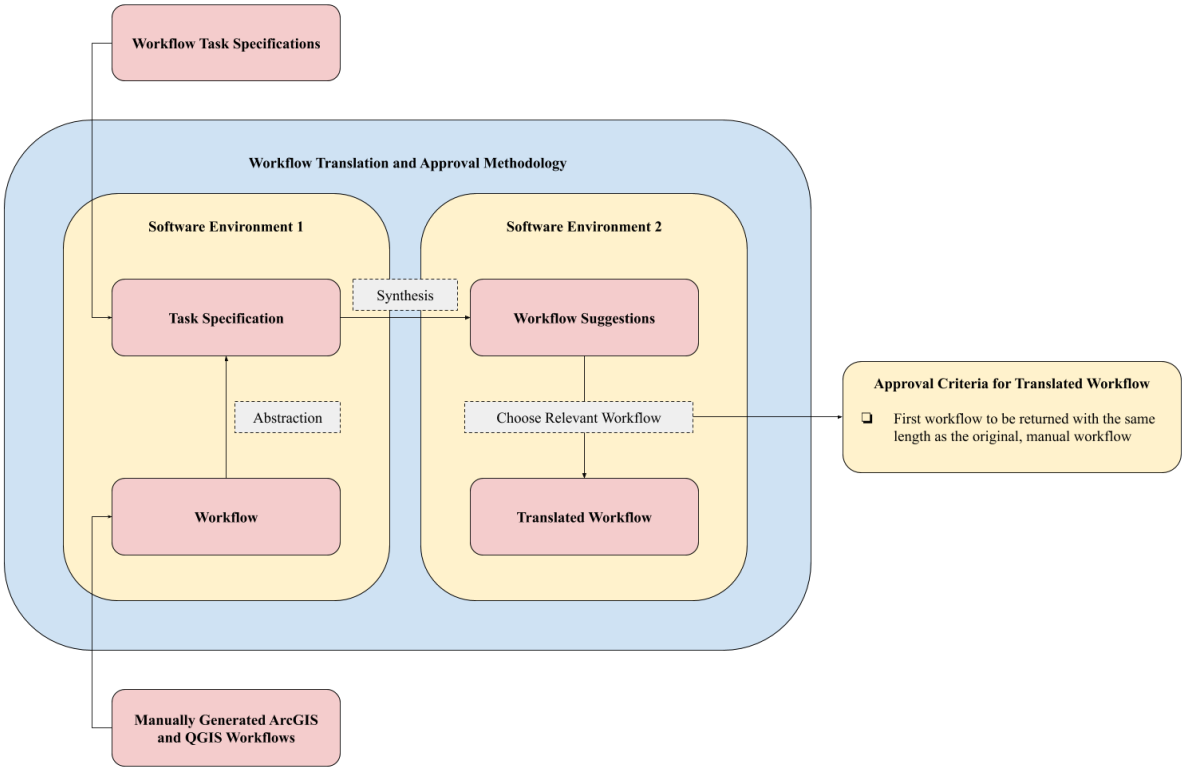
Firstly, the configuration of the synthesis engine requires that a GIS taxonomy based on the CCDT ontology, as well as a tool hierarchy which is simplistic in nature and only takes into account the sub-tool and super-tool combinations defined in the tool annotations, be created. The GIS taxonomy is provided in XML format which necessarily requires that the tools included in the tool hierarchy are also described in XML serialisation format rather than the RDF serialisation format they were originally conceived in when producing tool annotations. A clear difference in XML serialisation and RDF/turtle serialisation is that XML serialisation can only support a single data type as an input for a given tool. The nature of the tool annotations used, however, means that each unique combination of properties (core concepts types, geometric layer, data measurement levels) can be considered a unique datatype and, therefore, requires multiple data types as inputs. To circumvent this, the Python script used for translation of the annotated databases generate a unique data type for each included combination of data types. These can be considered to be unique leaves on a taxonomic tree (Meerlo, 2019). The GIS taxonomy is generated based on the implementation of a Python script, the basis of which is an OWL reasoning step which infers states about class combinations in the tool annotations. The development of this is beyond the scope of this thesis but Python script has been provided for use in this research. The implementation of the Python script to alter the GIS and tool taxonomies will likely be done several times as the tool annotations are refined over the course of the research.

Secondly, because the synthesis engine cannot directly read the RDF turtle serialisation, the ArcGIS and QGIS tool annotations are translated into JSON format by the same Python script implemented in the first step. When configuring the workflow, the maximum length was set at 10 and min was set at 1 and the total number generated for each was 20. This set of configuration parameters was set as they more than accommodate the variety of manual workflow specifications seen across the competency questions. Once these inputs have been sufficiently developed and prepared, they are used as inputs in the Java file on which APE is based and run to produce the relevant workflows. The APE file and the relevant configuration inputs used for this research thesis is available in a GitHub repository[5].

4.4 Workflow Translation

The fourth research stage is carried out as a series of sub-steps in a workflow translation mechanism by making use of the semantic elements developed and by implementing the already-configured Java based synthesis engine. These sub-steps make use of the relevant semantic elements as illustrated by the following figure (Figure 4.5).

Figure 4.5. Workflow Translation Mechanism



Firstly, the manually generated workflow for a given software environment is abstracted into a task specification in line with the competency question. Workflows are then synthesised using the preconfigured APE synthesis engine to produce a set of workflow suggestions for implementation in the alternative

[5] https://github.com/lexirowland/workflowtranslation

software environment. Workflow suggestions for each software environment are produced in set of 20 with a maximum length of 10. This maximum length was chosen to account for the range of potential lengths in the workflow set. Finally, to choose the final *translated workflow*, the set of workflow suggestions needs to be checked against translation approval criteria, which for the purpose of this research is simple (Table 4.4). Note that for the evaluation of the translation mechanism in the next stage, it is essential that the approval criteria remain independent of the evaluations. It is partly the reason for the simplicity of the approval criteria. This process is repeated for every competency question where ArcGIS workflows are generally translated into QGIS workflows using this translation mechanism but this, of course, can be repeated with QGIS as the origin workflow.

Table 4.4. Approval Criteria for Translated Workflow

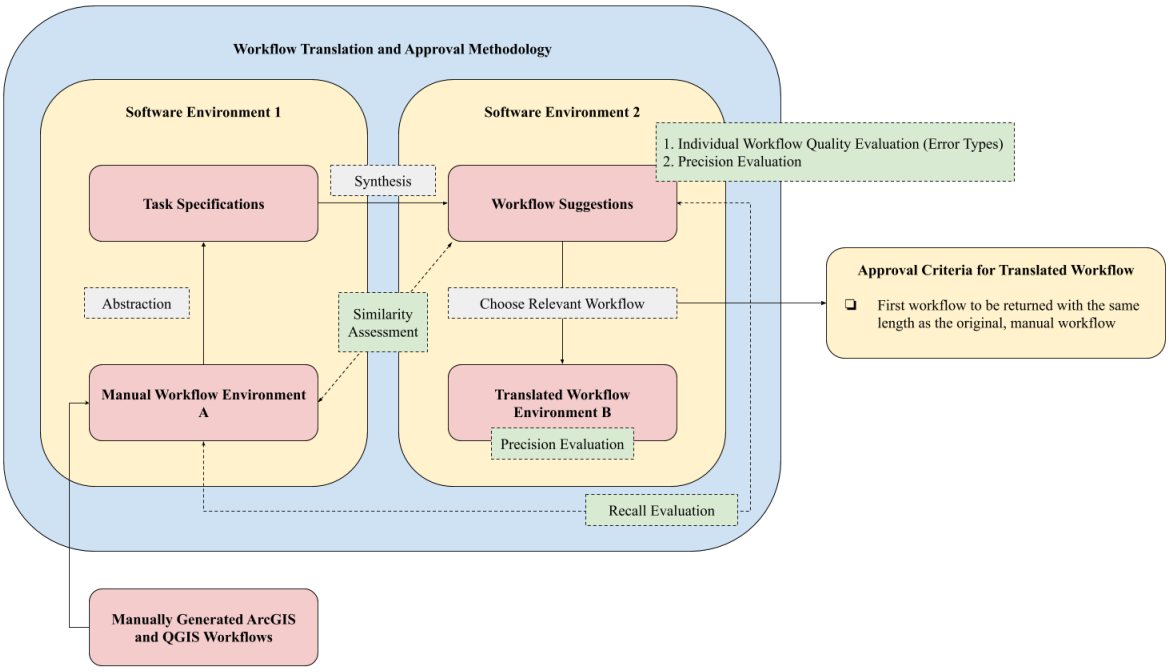
No.	Assessment Type	Description
1	Order	Workflows chosen as Translated Workflow need to be one of the first to be generated in the set and is dependent on the second criterion.
2	Length	Workflows chosen as Translated Workflow need to be of a similar length to the original workflow

4.5 Workflow Quality and Similarity Assessment

The final stage of this research will seek to undertake two assessments, namely, workflow quality evaluation and workflow similarity assessments. Workflow quality evaluation will take place in four sub-stages. Individual workflow quality of those workflows in the suggestion set for each competency question, the capacity of *the synthesis engine* to produce quality workflows based on recall and precision measures, the capacity of the ontology to support quality workflow translation/generation and the capacity of the *translation mechanism* based on a second precision metric. It should be noted that the first sub-evaluation will assess the absolute quality of the workflows on an individual basis, not the translation quality of the workflows which will be implicitly done through the similarity assessments and second precision calculation. The similarity assessment will be carried out to assess the similarity between the manually generated ArcGIS workflow and the translated QGIS workflow. This assessment will be carried out as an adaptation of Starlinger et al. (2014) structure-based approach to workflow similarity assessment.

The figure below illustrates which parts of the translation mechanism are involved in the respective quality evaluations and similarity assessments (Figure 4.6). Within the context of both the quality evaluations and the similarity assessments, there should be a delineation between absolute quality of a workflow as assessed in the quality evaluation and the assessment of translation quality. The latter assessment should take into account both quality of the workflows on an individual basis as well as the similarity scores they receive in order draw some conclusion as to their ability to reproduce domain knowledge across software environments. As such, conclusions regarding translation quality will only be made on completion of both the quality evaluations and the similarity assessments.

Figure 4.6. Quality Evaluation and Similarity Assessment in the Context of the Translation Mechanism.



4.5.1 Individual Workflow Quality Evaluation

The framework for individual workflow quality evaluation is taken from Kruiger et al. (2020) where translated workflows are evaluated based on the presence, or lack, or errors in the translated workflows. Identified errors in each workflow are categorised according to 4 categories of errors as outlined by the following table (Table 4.5).

Table 4.5 Error Types for Workflow Quality Evaluation (adapted from Kruiger et al., 2020).

Error Severity	Error Type	Definition
Hard	Signature	An error where a workflow or part thereof cannot be executed because a tool has been incorrectly applied to a workflow.
	Semantic Imprecision	This error tends to produce meaningless or invalid outputs for a given tool or competency question because the ontology is missing some semantic constraint on data.
Soft	Redundancy	Workflows make use of tool applications which are not essential in producing a valid output to the competency question.
	Data Quality	Error is seen where transformations in data have resulted in a reduction in geodata quality unnecessarily.

This individual workflow evaluation will be carried out manually by a GIS expert and the errors in each workflow documented. The identification and recording of these errors will be useful in subsequent research steps.

4.5.2 Synthesis Engine Quality Evaluation

Because a synthesis mechanism is central to the translation mechanism proposed in this research, it is essential that this is evaluated for its capacity to produce quality workflows based on the inputs provided. Here, workflows evaluated in the former step are placed into categories based on the error types they contain according to a confusion matrix (see Table 3.6). Because the synthesis mechanism will only produce workflows that it deems to be valid based on the achievement of the goal specifications, a false negative and true negative classification is not possible in this framework. However, a false negative can be identified where the expert-produced, manual workflow is not present in the set of suggested workflows generated. As such, the error types identified will allow for false positive, true positive and false negative categorisations. This limits the range possible metrics of classification used in this research to precision and recall as these measurements only make use of these three categories in its computation.

Once categorised, a precision calculation will be performed which calculates the proportion of predicted translated workflows were correct or valid. A fully functional synthesis engine should, therefore, have a precision measure of, at least close to, 100% for the mechanism to be deemed to produce quality workflows able to fully answer a given geo-analytical question. Additionally, a recall metric will also be calculated to assess whether the translation mechanism is able to find all correct possible workflows for a given competency question. Here, the manually translated QGIS workflow is used to check for a false negative within the workflow suggestion set which would only appear where the suggestion set does not contain a workflow which matches this manually translated workflow. A fully functional translation mechanism should, therefore, have a recall measures of 100%, particularly in the context of this research, to be deemed able to support the answering of a given geo-analytical question. The outcome of this metric will be either 0, completely lacking the ability to produce correct workflows based on a false negative, or 1 being able to produce correct workflows based on the presence of the correct, expert-derived workflow. Additionally, this calculation highlights whether the original expert-generated and optimal workflow is represented in the set of workflows generated for each competency question. This step is necessary to ensure that the proposed mechanism is capable to accurately reproducing the domain knowledge formalised as part of the mechanism (Bergmann & Gil, 2014) and identifying all correct solutions to a given geo-analytical question.

4.5.3 Ontology Evaluation for Workflow Translation

The capacity of the CCDT ontology to support quality workflow translations will be assessed using an error-based method which is an extension of the error type methods used to assess workflow quality on an individual basis. The functional evaluation of the CCDT ontology for workflow translation will follow the framework outlined by Kruiger et al. (2020) in his assessment of the capacity of the CCDT ontology to support quality workflow synthesis wherein a precision calculation is used to compute the capacity of the ontology. The evaluation of the CCDT ontology follows several steps as outlined below (Table 4.6).

Table 4.6 Ontology Evaluations Steps and Description

Step No.	Step Name	Description
1	Mechanism Calibration	The translation mechanism is calibrated to produce a set number of workflows (20) with a maximum length or number of tool inclusions (10). This is done in a previous methodological step.
2	Error Classification	Workflows are reviewed and any errors in the workflow are classified into the categories outlined in section 4.5.1. Again, this is done as part of a previous methodological step.
3	Precision Calculation	A precision calculation is implemented wherein the number of correct workflows based on GIS expert assessment is divided by the number of workflows in total, both erroneous and correct. This precision calculation is also done per error category to provide nuance.

As these steps highlight, this evaluation is not independent of the previous sub-evaluations. Indeed, it is based on the precision calculation performed in the previous evaluation but also includes the precision calculation across the individual error categories which provides nuance and allows for conclusions to be drawn as to the capacity of the ontology to support quality workflow translation.

4.5.4 Translation Mechanism Evaluation

The evaluation of the translation mechanism is performed based on a second precision metric calculation following the choosing of the Translated Workflow based on approval criteria. Due to time limitations, the Translated Workflow for each competency question is a QGIS workflow based on the translation from ArcGIS but could just as easily have been done in the reverse. This second precision calculation, which will be either 0 or 1, calculates whether the workflow deemed to the Translated Workflow is actually correct based on the review of an expert. Returning a precision of 1 highlights the ability of the translation mechanism as a whole is able to accurate produce a workflow with the quality necessary to fully answer a geo-analytical question. This second calculation is important as it is performed as an independent metric to the first precision metric and, therefore, serves to assess the translation mechanism as a whole rather than as a product the synthesis engine which has some short comings.

4.5.5 Workflow Similarity

The final methodological step, workflow similarity assessment, will assess the QGIS workflow suggestion set for similarity to the original ArcGIS workflow for a given competency question. Again, due to time limitations this was the direction of translation chosen but just as easily could have been performed in the reverse. While Starlinger et al. (2014) recommendations have guided the structure of this assessment, the assessment has been adjusted to be suitable for the GIS workflows and the components used in this research. These adaptations take two forms. Firstly, the annotation-based methods have been removed from this framework as these are not central and are not comprehensive enough for this research. Secondly, the central similarity measure used in this assessment is the Ratcliff-Obershelp similarity measure. Starlinger et al.'s (2014) structure-based approach is chosen as it has been argued to perform better than annotation-based

methods for the purpose of one-to-one workflow similarity assessment, even in the context of rich workflow annotations. It is based on the results of this similarity assessment and the assessment of the translation mechanism that conclusions as to the translation quality, rather than absolute quality, of the workflows produced by the mechanism can be made.

Step 1: Pairwise Module Comparison

The framework for similarity assessment has been slightly altered to ensure that the pair-wise module comparison carried out is more applicable to the datatype attributes assigned to each tool module. A data sequence similarity assessment based on the Ratcliff-Obershelp measure is used to make the pairwise module comparison. The Ratcliff-Obershelp measure works such that the longest possible sequence is found in the string and the process repeated iteratively to calculate similarity. The reason this is deemed most appropriate is because the tools are annotated based on their data types which follow a strict formatting structure, namely, spatial core concept, geometric layer type and then data type measurement. Because of this, a similarity based on sequences or 'chunks' in the data types rather than on individual characters is most appropriate as the latter has the potential to slightly skew the similarity score where a number of characters are similar but the sequence in which they are present does not necessarily matter. This measure is also used to find similarity between tool labels because there are several tools which have similar characters (see for example Zonal Geometry and Zonal Statistics). This measure of similarity has not previously been used for the comparison of workflows in this manner but, arguably, it provides a valid method of producing a pairwise module comparison due to the nature in which the tools are annotated in this research. The approach is implemented semi-manually but the computation of the Ratcliff-Obershelp measure itself is calculated by making use of a simple Python script developed for the purpose[6].

Step 2: Module Mapping

Following the pairwise similarity calculations, modules are mapped to each other based on the best similarity match between modules. Because the modules are mapped based on three characteristics, Starlinger et al. (2014) recommend that the module mapping also makes use of a maximum weight matching procedure. However, these three characteristics are closely associated with the individual tool that they are describing and should be seen as a tool bundle or package rather than individual characteristics of a workflow. In this case, it would not make sense to do a maximum weight matching and so this recommendation will not be used. Instead, matching will take place based on the shuffling of tools and the highest similarity score.

Step 3: Topological Workflow Comparison

Starlinger et al. (2014) gives three options for carrying out topological workflow comparison. Because the GIS workflows are, necessarily, direct graphs (DAGs), a topological comparison is required in this step. Indeed, when the shuffling of modules is done, a module mapping based solely on the highest similarity score such that no module mapping crosses each other may mean that the order in which tool modules appear now render the workflow invalid or meaningless for a given competency question. A topological comparison where order of tool appearance constrains the way in which module mapping can take place ensures that the validity of the workflow is secured. Assessing similarity in this way makes is known as the maximum weight non-crossing measurement where modules are mapped to each other only

[6]https://github.com/lexirowland/workflowtranslation/blob/master/similarity_assessment/ratcliff_obershelp_measure_automated.py.

where there is no crossing in the tool mapping and within this step there are two calculations to be made, namely, the additive similarity score for path pairs and the maximum non-normalised similarity based on path pairs.

This set of paths analysis is the most relevant to this research due to the nature of the workflows being compared. In the body of research discussed by Starlinger et al. (2014), workflows from the same software environments are compared for similarity as a method of information retrieval. Here, however, workflows are being compared for similarity across software environments where tool labelling conventions are different for similar tools. As such, the topological comparison needs to take place based on the order in which tool modules appear across workflows and based on implicit expert knowledge about the similarity of tools. Implementing a, for example, graph edit distance would not be sufficient in this case because editing the tool modules to be exactly similar to the ArcGIS workflow would be invalid as the resulting workflow would simply no longer be representative of the QGIS workflow. As such, this process remains a largely manual process.

Step 4: Normalisation

Although this step is an optional on in Starlinger et al.'s (2014) framework, it is required in the context of this research thesis due to the fact that the set of workflow suggestions generated by the translation mechanism will have differing lengths based on the use of super tools and potential tool redundancy. However, separate step for normalisation is not required in the context of this research. Indeed, the way in which the similarity score is calculated per workflow is by generating the score per intra-module characteristic and then averaging the score per module. The score per workflow is then an average of all the module scores in a workflow. Therefore, this normalisation already happens in the averaging of these module scores to produce a similarity score per workflow and normalisation following topological comparison is not required.

Chapter 5 Results

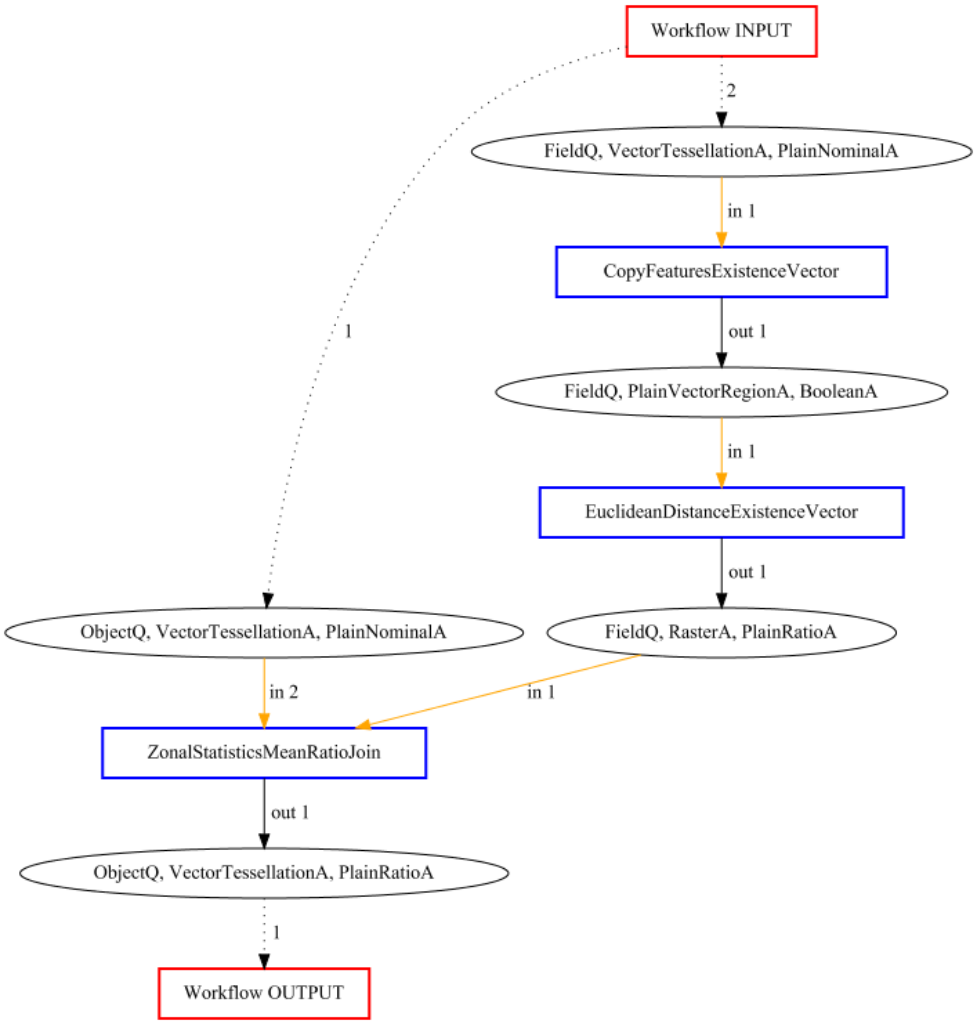
To assess the capacity of semantic technologies to support effective GIS workflow translation, three research goals needed to be achieved as part of the implementation of this research thesis. Firstly, a novel approach to the translation of GIS workflows using an ontology-based translation mechanism was defined. The outputs of the implementation of the translation mechanism will be presented in the first sub-section of this chapter and will include commentary as to effectiveness of making use of super- and sub-tools in the tool annotations. Preliminary commentary on the similarity of the workflows and the comparisons of tooling across the different software environments will also be presented. The result of second goal of this research, the evaluation of the quality of the translated workflows and, by extension, the quality of the synthesis engine, the translation mechanism and ontology used for this purpose, is presented and discussed in sub-section 5.2 of this chapter. Finally, the results of the assessment of similarity between the manual ArcGIS workflow and the final translated QGIS workflow with the goal of assessing functional similarity, as the third goal of this research, is presented in sub-section 5.3 of this chapter. The final subsection will discuss the success of this research project and make suggestions for further research.

5.1 GIS Workflow Translation Outputs

In addition to generating the taxonomy and transforming the RDF tool databases to JSON formats, it was also necessary to configure the engine for the conditions of each workflow. The maximum length of workflows constant value was chosen based on the fact that none of the manually generated workflows exceeded this value and the number of generated graphs were set at 20 in order to produce a large enough set that the analysis that followed would at least be based on a representative number of sample set for each competency question. The resulting workflows were generated in a distinct format, namely, begin state data types, tool, data type, tool, data type, tool, goal specification. The data types can be reused along the workflow because synthesis problem, i.e. the competency question, was specified to be a global problem. The results are available in both a text format and as an image as illustrated by the following figure (Figure 5.1).

This figure illustrates a close-to-optimal workflow for competency question 3 in the ArcGIS workflow. Here, the Coverage dataset representing the land use in Amsterdam is used as input to a Select By Attribute function based on a specific attribute, in this case Parks, and copied to an output existence vector dataset which is naturally at a Boolean measurement level. A Euclidean distance function is then performed to produce a raster dataset with a ratio measurement level, and this is used, in combination with Lattice dataset representing neighbourhoods in Amsterdam in the Zonal Statistics function and then joined back to the Lattice of the neighbourhoods in Amsterdam. Both the Select by Attribute and Copy tool, named here as “CopyFeaturesExistenceRaster”, and the Zonal Statistics and Join tool, named here as “ZonalStatisticsMeanRatioJoin”, are examples of the super tools used as part of this research to improve workflow quality.

Figure 5.1 Example Workflow from the Set of Generated ArcGIS Workflows for Competency Question 3



The novelty and difficulty seen in generating some of the workflows used in this research is discussed in the following sub-sections.

5.1.1 Presence of Super- and Sub Tools in Translated Workflows

Part of the novelty of this research is the use of super tools in the tool databases in order to both improve some of the logic behind the annotation of specific tool combinations and to reduce potential, and sometimes unnecessary, complexity seen in the translated workflows. Because the synthesis engine places tools together based on their matching data inputs and outputs, annotating these tool combinations in the database as a super tool allows for the enforcement of certain tool combinations and reduces the potential for non-logical tool combinations to be present in the translated workflows. Indeed, a super tool can be seen as a type of constraint on the workflow and has the potential to increase the quality of the workflows produced by the translation mechanism. This research identified a number of relevant super tool combinations across both software environments. Whether the quality of the workflows was effectively raised using super tools in this research will be presented in the following subsection (see Tables 5.3 and 5.4).

Sub tools have been used in this research in order to ensure that valid measurement level inputs are semantically defined in the tool databases. Indeed, it is true that certain tools require a data type with a specific measurement level, some can use a number of data types with a range of measurement levels where the input measurement level defines what the outputs measurement level is and others it is not necessary to specify a measurement level because the tool takes as input all measurement levels and/or geometric layer types. For example, Thiessen Polygons in ArcGIS will always make use of Point Measures as a valid geometric layer type and Coverage will always be a valid output. However, using Point Measures with a ratio measurement level must always result in a Coverage with a ratio measurement level and the same is true for Point Measures with an interval measurement level which should always result in a Coverage with an interval measurement level. To accurately annotate the tools to specify all these combinations of parameters, layer types and measurement levels, sub tools should be used for each tool where these combinations are found. This use of sub tools, again, act as a constraint which effectively increases the quality of the workflows produced by the synthesis engine because valid tool combinations become more likely (Meerlo, 2019).

5.1.2 Difficulties in Implementation

In formalising the database and configuring the synthesis engine, there were a number of difficulties faced. Firstly, although there were a number of equivalent and almost equivalent tools between ArcGIS and QGIS, there were some tool examples which required further research to map across the environments and a number of tools which do not have some sort of equivalence across the environments. In some cases, the development of a super tool was possible to generate some semblance of similar tool in the alternative environment. For example, in ArcGIS there exists a tool Euclidean Distance which can take a Field Raster as an input and calculate the Euclidean distance to a specified source. In QGIS, this tool does not exist but can arguably be overcome with the combination of rasterising an Object Vector data source and then performing a proximity analysis and the combination of these tools can be formalised in the QGIS database as a super tool. Other tools, however, have no equivalent in QGIS; of which areal interpolation is arguably one. This will be further discussed in section 5.3.

Secondly, during the first attempts at running the synthesis engine to generate the translated workflows in each environment, the set of generated workflows, when looked at closely, highlighted some minor issues with the way in which tools had been formalised in the database. For example, a workflow requires a specific input, say a point object at nominal measurement level, either no workflows were generated by the engine or the set of workflows generated were not at all valid or similar to the expert generated workflows. In general, this points to an issue with the way in which tools were formalised in the database such as a lack of specificity in the annotation of the tools relating to measurement levels or an absence of potential tool combinations altogether. This issue is fixed through an iterative process of identifying where invalid tools combinations are being seen and checking whether this is due to an issue with the annotation of the tools. This process ensures that workflow errors due to annotation errors are limited. Every time a database is changed, the GIS taxonomy and transformation into JSON needed to be done in order to correctly configure APE engine and, as such, particularly where there were a number of issues with the databases, this proved to be a time consuming process.

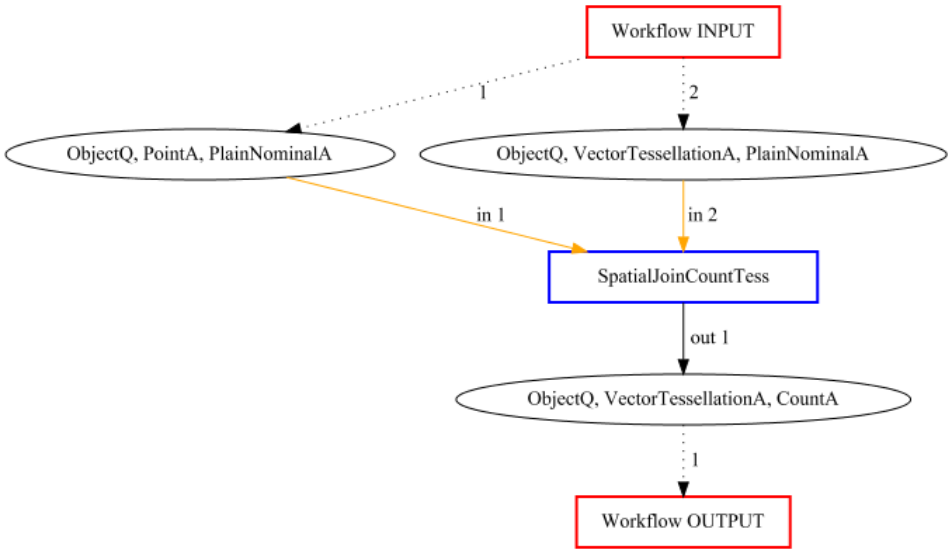
5.2 Quality Evaluation

The evaluation of workflow quality took place in three sub-stages. Firstly, for each question, the set of 20 generated workflows for each software environment was assessed for their individual quality by evaluating whether errors were present in the workflows. Errors are categorised into either hard or soft errors with each category having 2 subcategories. Secondly, the workflow synthesis mechanism was assessed for its ability to produce quality workflows. This assessment is a direct result of the previous substage and is based on precision and recall computations following workflow categorisation. Finally, the capacity of the CCDT ontology for workflow translation will be assessed; again, making use of the error-based method of assessing workflow quality.

5.2.1 Individual Workflow Quality Evaluation

As part of this substage, several workflows will be presented and discussed based on the errors, or lack thereof, seen in the workflow. The following workflow (Figure 5.2) was the first workflow generated in the set of ArcGIS workflows suggestions following the specification of the inputs and outputs for the first competency question. Here, the workflow makes use of the spatial join function with count sub-operation which takes as input the specified workflow inputs, both at nominal measurement levels and produces a lattice at count measurement level which counts all the medical facilities in each neighbourhood of Amsterdam and returns them as a lattice with an attribute table. This is the most optimal workflow for answering this competency question and has no workflow errors. The workflow presented above (Figure 5.1) is also an example of a workflow which is an optimal workflow in the ArcGIS environment and, as such, has no workflow errors.

Figure 5.2 ArcGIS Workflow Solution to Competency Question 1



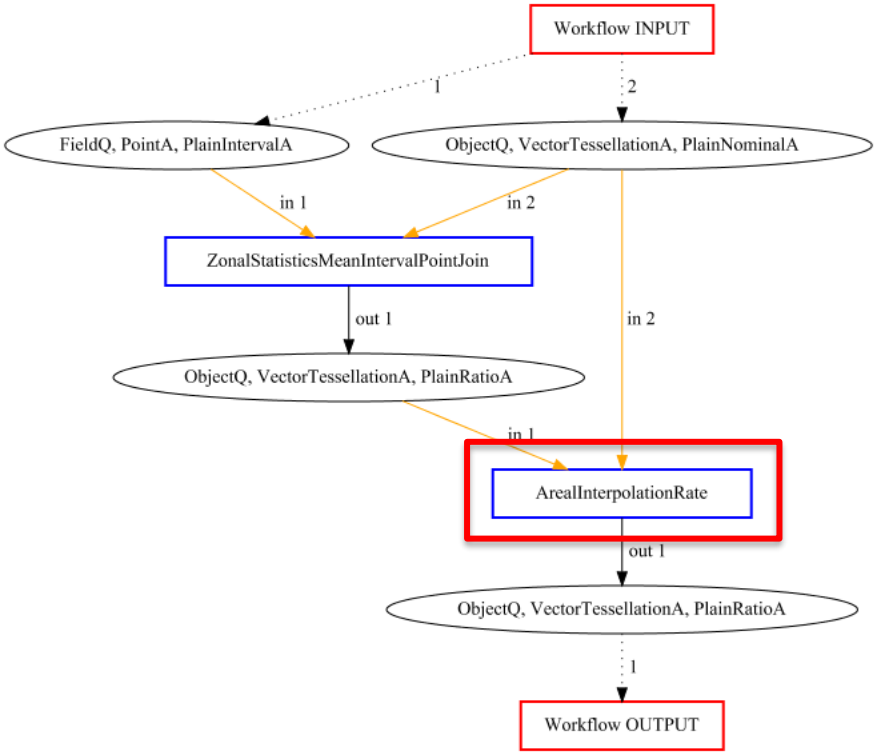
In the tables that follow, the soft errors are reported for a workflow only when the workflow did not contain any hard workflow errors. Because the synthesis engine was able to identify the most optimal workflow in some cases, the sum of the hard and soft errors is not equal to the number of workflows but the sum of the hard errors for a set of generated workflows is the inverse of the sum of the number of correct workflows.

Indeed, if the workflow only contains soft errors, the workflow was deemed correct and its errors counted where necessary.

a. Workflow Hard-Errors

Hard workflow errors are defined as errors which render the workflow invalid or unable to meaningfully answer the competency question it was generated for due to the use of a tool or tool combination in a workflow that has either been incorrectly applied (signature error) or because the output that has been produced from a particular tool is invalid or meaningless for a given competency question due to the ontology missing some semantic constraint (semantic imprecision). In general, invalid outputs for a competency question, as a whole, will not be seen in the set of generated workflows because a goal for each question has been defined. Indeed, invalid workflow outputs will result in the workflow not being generated at all but it is possible a tool or tool combination will produce an output in a workflow that it should not have and will, therefore, result in semantic imprecision. For a workflow to answer a spatial question, the workflow, necessarily, cannot contain a hard error. Following the manual evaluation of quality for each workflow generated for each competency question, amounting to a total of 200 workflows for both software environments. The following illustrates some examples of hard errors seen in the generated workflow set for both software environments.

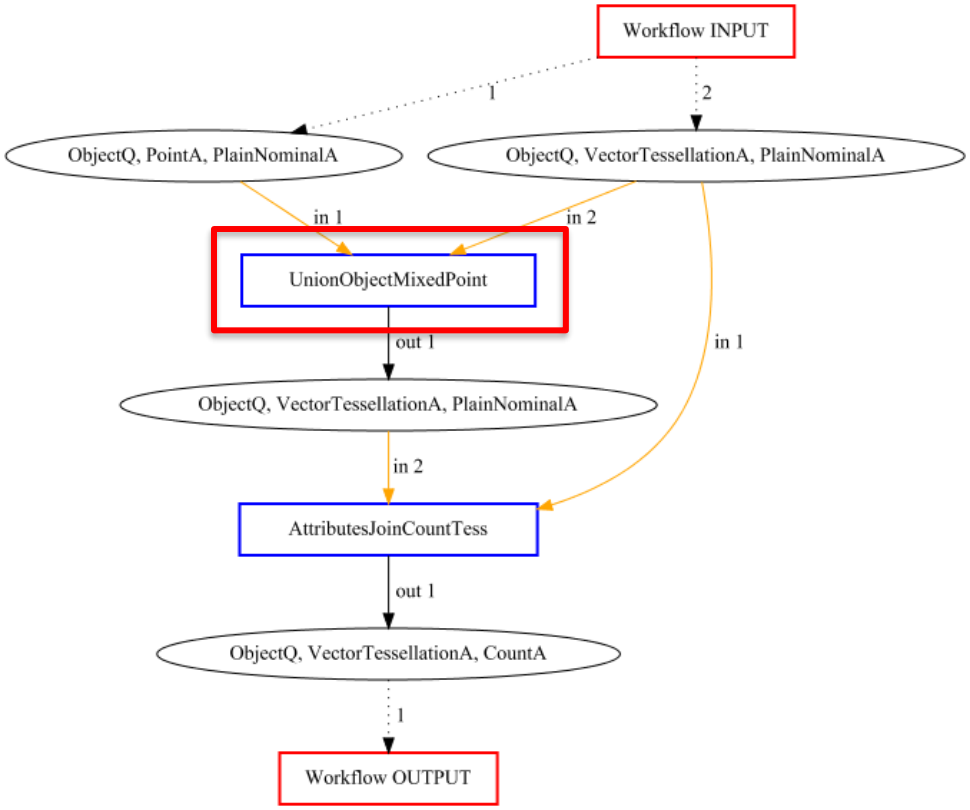
Figure 5.3 Example of a Semantic Imprecision Error (ArcGIS competency question 5)



The workflow above, generated as a solution to competency question 5, makes use of two inputs, namely, a point measures dataset which represents the temperature reading around the municipality of Amsterdam for a given period. This is dataset type is formalised by the ontology as a field dataset with point attributes. The second input is a lattice dataset representing the polygons of the neighbourhoods in Amsterdam. These

two inputs are used by a super tool combination of ArcGIS's zonal statistics and join where an average of the temperature within each zone or neighbourhood is produced by the zonal statistics mean function and then the output joined to the lattice of the neighbourhoods. Following this, the workflow then initiates an areal interpolation operation which is supposed to be applied where there are two polygon datasets which differing boundaries or demarcating different areas and the input set of polygons is used to make predictions about the data values for a new set of polygons based on their overlap in location. In this case, this is an incorrect application of the tool because the polygons being used are both a lattice of the neighbourhoods in Amsterdam and, as such, produces a meaningless result. The annotation used for areal interpolation is missing the semantic constraint that the polygons need to be representing different phenomena and have different boundaries for this to be an applicable tool.

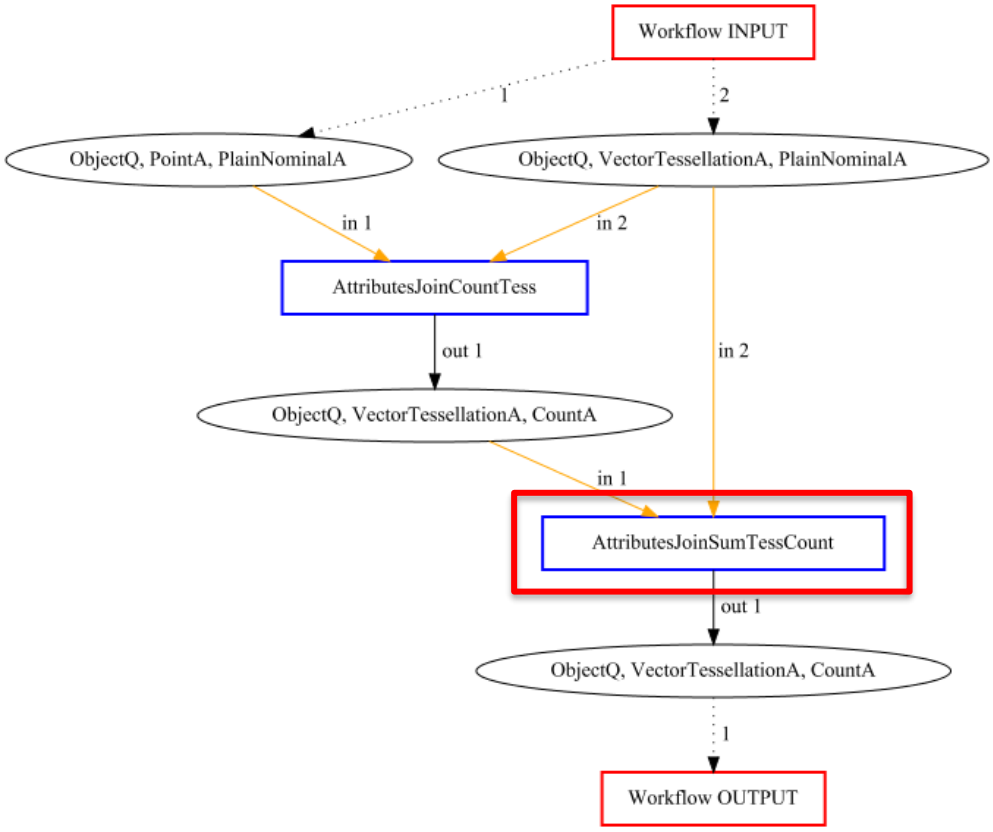
Figure 5.4 Example of a Semantic Imprecision Error (QGIS competency question 1)



The workflow above is presented as a QGIS solution to competency question 1. Here, QGIS's union tool uses a point dataset representing all the medical facilities in Amsterdam as well as the lattice dataset containing polygons with all the neighbourhoods in Amsterdam as input. The union tool then produces a lattice as output were the points are transformed into small polygons and surrounded by irregular shaped polygons and, where points come into contact with the original polygons, they receive the polygon attributes and vice versa. Following this, a spatial join operation making use of the count sub-operation wherein the polygons produced from the union output are joined and the union polygons counted based on their overlapping location with the original lattice representing the neighbourhoods of Amsterdam. While the union tool and its output can be a valid tool for a particular kind of workflow, therefore making the semantic constraints valid, it does not make sense to apply this for this particular workflow and, as a consequence,

nor does the following tool or its output make sense and therefore renders the output meaningless due to semantic imprecision.
The following workflow, presented as a QGIS solution to competency question 1, shows an example of a signature error which renders the workflow invalid and cannot be executed. Here, a point dataset representing all the medical facilities in Amsterdam and a lattice dataset representing all the neighbourhoods in Amsterdam is used as input for a spatial join operation with a count sub-operation. Here, the number of medical facilities in each polygon are counted and the output lattice dataset contains this attribute for each neighbourhood of Amsterdam. This output is then used for a further spatial join operation which makes use of a sum sub-operation with data at count measurement level. However, the lattice dataset is at nominal data measurement level which means the attribute being used in this case is delivered as a string. String data types cannot be used in a sum operation and this renders the workflow invalid because this operation cannot be completed. As such, this is an example of a signature error in a workflow.

Figure 5.5 Example of a Signature Error (QGIS competency question 1)



It should be noted that the synthesis engine was completely unable to return a correct workflow for competency question 2 for the QGIS workflow environment. Here, each workflow contained some hard error, which for the most part, placed the necessary tools for the workflow together but the combination was unable to provide a completely meaningful output for the competency question. This may have to do with the fact that no workflows with the correct length required for validity were returned and perhaps the calibration for this question should have been changed to force longer workflows to be generated. This may also explain why ArcGIS for the same competency question performed poorly too. The complete

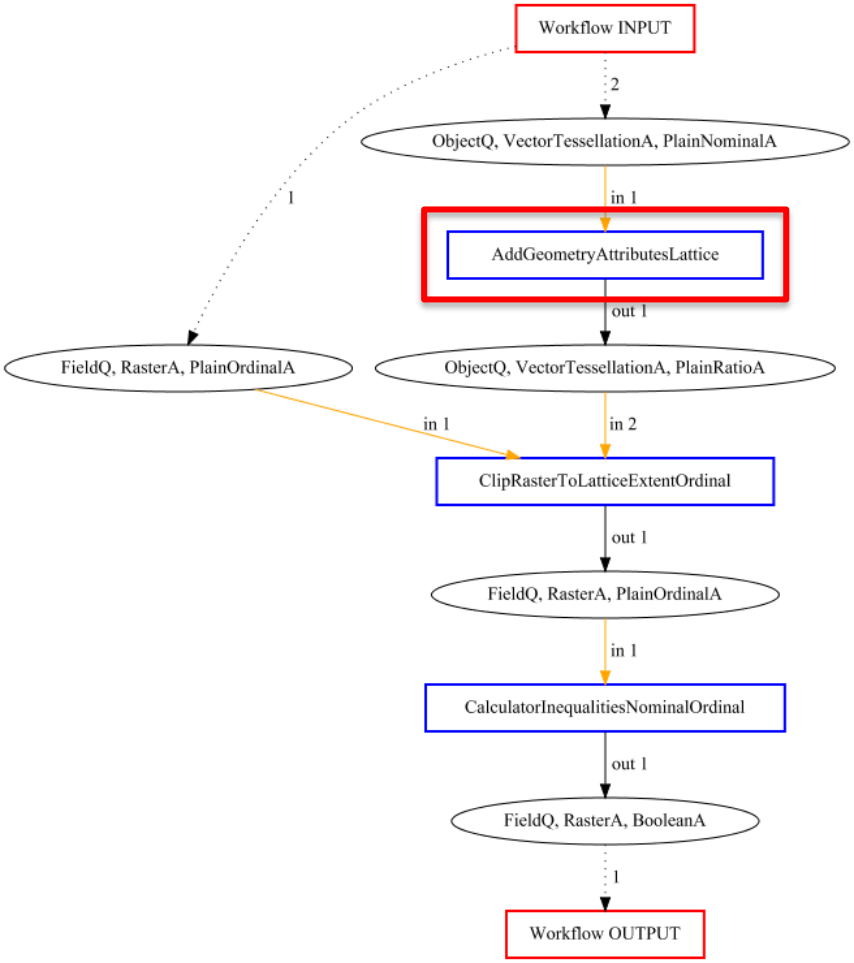
breakdown of both the hard and soft errors identified in each generated set of workflows for each competency question in each software environment is provided in Table 5.2 (below) where a full discussion on the hard errors seen is included.

b. Workflow Soft-Errors

The second type of workflow error that can be identified is a soft workflow error which has the effect of reducing the quality of the workflow or a less efficient solution but does not render the workflow incapable of answering the given competency question. In a similar manner to the hard workflow errors, each workflow for each competency question was evaluated for soft workflow errors. The following provides some examples of these identified and then summarises all errors found for each competency question in each software environment (Table 5.2 below).

The following workflow, generated as a QGIS solution to competency question 4, adds a geometry attribute to the lattice input dataset which represents all the neighbourhoods in Amsterdam. This lattice dataset is then used as input, along with the contour dataset representing the noise levels across Amsterdam, for the ClipRasterToLatticeExtent tool which serves to clip the contour to the extent of the Amsterdam neighbourhood of interest for this question. Following this, a raster calculator function is implemented where the output is an existence raster depicting the locations in Amsterdam which have a noise level greater than 70dB. As is clear, the initial addition of an area attribute to the neighbourhood lattice was no necessary for this workflow to give a valid output and, therefore, illustrates a redundancy workflow error.

Figure 5.6 Example of a Redundancy Workflow Error (QGIS competency question 4)



In Kruiger et al.'s (2020) work, it was noted that the number of redundant and quality errors seen in workflows could potentially be reduced should the measurement levels be added to the specification of inputs and goal outputs for the workflows. The lack of data quality errors in this analysis speaks to this sentiment. However, redundancy in some of the generated workflows for some competency questions is still high. Arguably, the reason for this is because the synthesis engine looks to synthesise exactly 20 workflows for each competency question in each software environment based on the configuration done in the context of this research. To meet this criteria, the synthesis engine arguably makes use of unnecessary tools and, therefore, creates redundancy. Constraining the data type inputs on these redundant tools to a more extensive degree may be a solution to this, however, the tools which are often identified as redundant are those which do not have complicated inputs and outputs and, therefore, the data measurement levels are not effective in reducing this redundancy. For example, the redundant tool seen in the workflow above is an add geometry attribute tool where a field is added in an attribute table (for which a measurement level is not needed) and then an area for each polygon calculated to produce a ratio scaled attribute as output. If a measurement level is added to the tool's annotation to reduce this, it would need to add every possible measurement level because the attributes in a data layer are not being changed by this function. This, therefore, does not sufficiently constrain the tool to reduce redundancy and so this process would be futile. It is argued that more semantic detail would need to be added to tools such as these, the focus of which is

designated as future research and is beyond the scope of this research. It should be noted that this reduction in soft workflow errors may also be due to the fact that an newer version of the synthesis engine was used in this research versus the one used in Kruiger et al.'s (2020) research.

The following table (Table 5.1) illustrates the breakdown of workflow errors identified across all competency questions and both software environments. There are several issues to note and explore. Firstly, competency question two and three produced the highest number of hard workflow errors in both software environments. In both environments, this is primarily due to the appearance of invalid sub-operations (sum and count) for either spatial join operations or the zonal statistics tool which resulted in valid data type combinations across the workflow based on semantic constraints but not a meaningful answer to the competency question. Without a method for further semantic constraints of these sub-operations, this will remain the case. For the same competency question, an existence vector was also often transformed into a field point or line dataset. Again, this is semantically correct but meaningless for our purpose and, therefore, renders the workflow invalid in this context. Secondly, the proportionately higher level of semantic imprecision for QGIS solutions to competency question three is a result of the fact that QGIS does not have a strict areal interpolation tool available and will naturally reduce the individual quality of any solution available for this competency question; an issue which will be discussed in the analysis of similarity in the following subsection. Lastly, redundancy, as discussed above, remains a problem with synthesis of workflow despite the upgrades in the engine and with the inclusion of constraints related to data measurement. This is reflected in the fact that the levels of redundancy seen in the table below is almost identical to the levels of redundancy seen in Kruiger et al. (2020) for workflow synthesis using the same ontology.

Table 5.1 Breakdown of Identified Errors in ArcGIS Workflows Generated (with Super Tools)

				Hard Errors		*Soft Errors*	
Question	**Software Environment**	**Workflows**	**Correct**	**Signature**	**Semantic Imprecision**	**Redundancy**	**Data Quality**
1	ArcGIS	20	15	2	3	14	0
1	QGIS	20	12	1	7	11	0
2	ArcGIS	20	4	7	9	3	0
2	QGIS	20	0	3	17	0	0
3	ArcGIS	20	4	0	16	0	0
3	QGIS	20	5	0	15	1	0
4	ArcGIS	20	13	0	7	13	0
4	QGIS	20	17	0	3	17	0
5	ArcGIS	20	15	0	5	13	0
5	QGIS	20	19	0	1	17	0
1-5	ArcGIS	100	51	9	40	43	0

1-5	QGIS	100	53	4	43	46	0

c. Super Tools and Workflow Quality

Although not the primary focus of this research project, the effectiveness of the novel use of super tools to improve workflow quality was of interest in this phase of research. To limit the amount of time spent on this particular issue, the difference in the individual quality of workflows generated for the implementation of QGIS was not investigated. However, it is argued that the results below certainly highlight the fact that the use of super tools in the development of annotated tool databases is necessary to raise the quality of the output set of workflows. The individual quality of ArcGIS workflows was assessed first for the set of ArcGIS workflows where super tooling was present, but annotations of individual tools were also present within the annotated database.

Table 5.2 Breakdown of Identified Errors in ArcGIS Workflows without Strict Super Tooling

				Hard Errors		*Soft Errors*	
Question	**Software Environment**	**Workflows**	**Correct**	**Signature**	**Semantic Imprecision**	**Redundancy**	**Data Quality**
1	ArcGIS	20	14	4	2	13	0
2	ArcGIS	20	2	5	13	0	0
3	ArcGIS	20	4	4	12	0	0
4	ArcGIS	20	13	0	7	13	0
5	ArcGIS	20	11	5	4	7	0
1-5	ArcGIS	100	44	18	38	33 (75%)	0 (0%)

Secondly, the annotations of individual tools which form part of a super tool were removed from the annotated database and the configuration of the synthesis engine was repeated to generate a new set of workflows. The quality of individual workflows for each competency question with this limited database is summarised in Table 5.3 (below).

Table 5.3 Breakdown of Identified Errors in ArcGIS Workflows With Strict Super Tooling

				Hard Errors		*Soft Errors*	
Question	**Software Environment**	**Workflows**	**Correct**	**Signature**	**Semantic Imprecision**	**Redundancy**	**Data Quality**
1	ArcGIS	20	15	2	3	14	0
2	ArcGIS	20	4	7	9	3	0
3	ArcGIS	20	4	0	16	0	0
4	ArcGIS	20	13	0	7	13	0

5	ArcGIS	20	15	0	5	13	0
1-5	ArcGIS	100	51	9	40	43 (84%)	0 (0%)

As is illustrated by the comparison of these two summaries, the number of correct workflow generated in the output set of workflows across competency question increases from 44% to 51% with the strict use of super tools. More interestingly, however, is the reduction in the signature errors where workflows are rendered invalid because tools are incorrectly applied in a workflow. The super tools work to reduce the appearance of this error by forcing logical connections between tools within the annotated database and, therefore, ensuring they appear together as one tool in a workflow to produce the correct output. Indeed, the super tools reduce the number of appearances of tools which are necessary in a workflows to produce a meaningful output but do not make sense when they appear on their own and tend to result in the workflow being invalid. The ArcGIS tool "Copy Features" is an example of this because it cannot be run without the inclusion of a "SelectFeaturesByAttributes/Location" tool preceding it. Additionally, there is a slight increase in semantic imprecision and redundancy with the strict use of super tools, although only a small difference per competency question, which generally results from the synthesis engine forcing tool combinations in order to generate exactly 20 workflow solutions and points to the difficulties in using the synthesis engine under this specific calibration rather than with the use of the super tools themselves as discussed above. Moreover, the set of workflows generated for each competency question where super tools were used were also more similar to each other in terms of the tools they featured than where super tools were not used. Indeed, where super tools were used strictly, the difference between workflows for a single competency question tended towards differences in the sub-operations of the same tool rather than making use of a different tool altogether. This, on the one hand, is promising because the correct tools are being chosen by the synthesis engine and there are less random tools being included but there is an increase is signature and semantic errors due to the difficulty in constraining the sub-operations correctly.

5.2.2 Synthesis Engine Evaluation

To evaluate the synthesis for its capacity to answer the spatial questions, the workflows generated for each competency question are sorted into four categories based on the error type. This categorisation is then used to assess the precision and recall capabilities of the synthesis engine and, therefore, assesses the ability of the engine to produce workflows of a high quality and identify all relevant workflows for a given competency question.

a. First Precision Calculation

Precision is used as a measure which denotes what proportion of predicted translated workflows were correct or valid. Workflows where hard errors were identified were designated as false positives and all other workflows, including those which had soft errors, were designated as true positives. If the precision metric returns a value of 0, then not a single workflow was generated that did not have an error of the given type. If the precision metric returns a value of 1, then every workflow that was generated did not have a single error of a given type and was deemed to be a true positive within the specified error category. The breakdown of the precision metric for hard errors is shown in the table below (Table 5.4). What should be noted as the analysis is presented below is that the synthesis engine generally performed in a similar manner in terms of its precision and recall and, indeed, in a similar manner to previous research in producing

workflows in both software environments. This highlights the ability of the engine to support quality workflow output in a range of environments.

Table 5.4 Hard Error Precision for Both Software Environments

Software Environment	Number of Workflows	Precision: Signature	Precision: Semantic Imprecision	Precision: Hard Error
ArcGIS	100	0.92	0.60	0.51
QGIS	100	0.96	0.57	0.53

There are several discussion points in the above table. Firstly, the summary precision metric for signature errors denotes that almost all workflows generated did not have a signature error. This lack of signature errors across the competency questions and software environments reflects the ability of the synthesis engine to correctly apply tools based on their data type semantics and is, arguably, supported using super tools. Secondly, the precision metric related to semantic imprecision denotes that 39% and 43% of workflows generated by the synthesis engine across ArcGIS and QGIS environments respectively were actually invalid because the tools were applied in such a way that they produced an invalid or meaningless output for the given competency question. This, of course, means that the synthesis engine is still, to a large degree, lacking the ability to produce meaningful, quality workflows for competency questions. As discussed above, this is likely due to the lack of more specific semantic constraints with regards to sub-operations as well as with regards to the forced generation of 20 workflows for each competency question. Finally, the precision metric for hard errors overall highlights that just under half of the workflows generated are considered invalid across the competency question due to the presence of hard errors. The amount of hard errors present in the workflows is high and, therefore, the ability of the engine to produce quality and meaningful solutions to the competency questions in this context is limited.

Table 5.5 Soft Error Precision for Both Software Environments

Software Environment	Number of Workflows	Precision: Redundancy	Precision: Data Quality	Precision: Soft Error
ArcGIS	100	0.57	1	0.57
QGIS	100	0.54	1	0.54

The workflows that did not have hard errors were subsequently analysed for the presence of soft errors. Of all the ArcGIS and QGIS workflows generated for all competency questions, including those that were identified as having hard errors, 57% and 54% of these respectively did not have a single redundancy error. Because none of the workflows generated had data quality errors because of the level of semantic detail with regards to data measurement levels of inputs, goal specifications and tool inputs and outputs, the same percentage of the workflows did not contain soft workflow errors. However, this precision calculation should be further nuanced to provide the precision metric in the table below. Here, the precision metric for the soft workflow errors are calculated following the removal of those workflows for a generated set which contain hard workflow errors. As such, the total number of workflows is reduced to only those which do not contain hard workflow errors and, therefore, provides the percentage of workflows that did not contain

a single soft error with respect to those workflows which are deemed to be correct but may contain soft errors. This is a more meaningful metric for this analysis. Additionally, the percentage of workflows which did not contain a single workflow error, both soft and hard errors, out of the entire generated set is denoted in the final column of Table 5.6.

Table 5.6 Total Error Precision for Both Software Environments

Software Environment	Precision: Hard Error	Precision: Soft Error (Not Including Invalid Workflows)	Precision: Any Error
ArcGIS	0.51	0.16	0.08
QGIS	0.53	0.13	0.07

To answer the competency question in the most efficient way, the workflow, necessarily should not contain any redundant tools or reduce any data quality unnecessarily. For ArcGIS, the synthesis engine produced a soft workflow error in 71% of the workflows which were analysed and for QGIS this redundancy was higher at 80%. It is argued that this redundancy is in part due to the forced creation of exactly 20 workflows as part of the calibration of the engine. Interestingly, if we apply the presence of workflow errors in the strictest fashion, only 8% of the ArcGIS workflows would be deemed valid and only 7% of the QGIS workflow would be deemed valid in this respect. As such, the ability of the synthesis engine used in the translation mechanism to produce completely valid and quality answers to spatial questions is seemingly limited. However, nuance should be taken in these metrics. Because the calibration of the engine used in the translation mechanism is as such, redundancy and semantic imprecision is common because the engine is aiming to find 20 workflows based on goal specifications and tools are added at random to achieve this goal. Because this redundancy is high, it would seem that the engine is limited in its ability to produce meaningful and valid outputs. However, for each workflow there is one or two optimal approaches to answering the spatial question before redundancy or data quality is naturally introduced to achieve 20 workflows in a set and whether or not this optimal workflow is seen in the set does not play a significant role with the precision metric calculation. In this case, the recall metric may be used to add some nuance to this discussion.

b. Recall Calculation

Recall is used a measure to assess whether the synthesis engine can find all correct workflows for a given competency question based on expert judgement. This calculation highlights whether the original expert-generated and optimal workflow is represented in the set of workflows generated for each competency question. As such, a fully functional synthesis engine should, therefore, have a recall measures of 1 or 100% to be deemed able to support the answering of a given geo-analytical question. The results of this metric are expected to be either 1 or 0. The following table (Table 5.7) summarises the total workflow errors, workflows with no errors and false positives present in workflows for both software environments.

Table 5.7 Breakdown of Relevant Inputs for Recall Metric per Competency Question and Environment

Question	Software Environment	Workflows	Correct	Total Hard Errors	Total Soft Errors	No Errors	False Negatives
1	ArcGIS	20	15	5	14	1	0

1	QGIS	20	12	8	11	1	0
2	ArcGIS	20	4	16	3	1	0
2	QGIS	20	0	20	0	0	1
3	ArcGIS	20	4	16	0	4	0
3	QGIS	20	5	15	1	4	0
4	ArcGIS	20	13	7	13	0	1
4	QGIS	20	17	3	17	0	1
5	ArcGIS	20	15	5	13	2	0
5	QGIS	20	19	1	17	2	0
1-5	ArcGIS	100	51	49	43	8	1
1-5	QGIS	100	53	47	46	7	2

The following table (Table 5.8) summarises the recall metric per software environment. As is clear, for both software environments, the synthesis engine used in the translation mechanism developed for this research is able to recall approximately 97% (average of both environments) of all correct combinations of tools which result in valid workflows. Although sufficient for the purpose of this research, this metric should be used with caution because it is impossible to know whether there may have been other valid combinations of tools based on their semantics which were not known to the GIS expert during manual evaluation. If looked at another way, however, the synthesis engine identified the optimal workflow in each competency question in both software environments 70% of the time which is also a positive indication of the recall capacity of the synthesis engine in its function within the translation mechanism. This recall metric, as suggested above, provides the evidence that while the synthesis engine is error prone, it is, indeed, able to produce the optimal, quality outputs needed for meaningful answering of spatial questions and is a well-functioning part of the translation mechanism.

Table 5.8 Total Recall Metric for Both Software Environments

Software Environment	Number of Workflows	Number of Correct Workflows	Number of False Negatives	Recall Metric
ArcGIS	100	51	1	0.98
QGIS	100	53	2	0.96

5.2.3 Ontology Evaluation for Workflow Translation

The methodological steps to evaluate the ontology for its capacity to reproduce the domain knowledge in supporting workflow translation have been indirectly carried out through the evaluation of other components related to this research. By comparing the precision and recall metrics as well as the individual workflow quality metrics, it is possible to evaluate the functional capacity of the CCDT ontology to

reproduce the domain knowledge. Firstly, the CCDT ontology, by its very nature, was designed to make connections between GIS tools based on their data types. This research has shown that the ontology is effective in doing this, with importance placed on the fact that it was able to do this across two software environments by abstracting tools to their data types and allowing for workflows to be translated with a fairly large degree of success. However, because the ontology was not specifically designed for the answering of spatial questions, the presence of hard workflow errors is still fairly common (57% and 51% for ArcGIS and QGIS respectively), particularly where sub-operations such as sum, median, mean are not yet entirely constrained by data types alone. Indeed, although the role of the ontology in aiding the answering of spatial questions, particularly across a range of software environments, should not be undermined, the ontology itself is still a far from a complete representation of all geo-analytical knowledge required to improve the quality of the workflows being generated across both environments.

The fact that the recall metric is as high as it is does reflect the fact that ontology is able to support the generation and translation of workflows across a range of environments sufficiently. Secondly, the presence of soft workflow errors should not necessarily be taken as an indication of the lack of capacity for the ontology to support the replication of domain knowledge through workflow synthesis and translation. Rather, the shortcomings of the synthesis engine itself when used in this context should be considered. Indeed, when forcing the engine to produce exactly 20 workflows, unique tool combinations are more common but so is the potential for redundancy. Without the ability to constrain tools beyond their data types, the ontology cannot effectively reduce this and perhaps suggests the use of an alternative ontology for this purpose. As a strict data types ontology, however, the CCDT ontology can sufficiently support the generation and translation of tools across multiple software environments.

5.2.4 Translation Mechanism Evaluation

Following the selection of the Translated Workflow, which for ease of readability is presented in the following section, a second precision metric is calculated to assess the capability of the translation mechanism to produce quality workflows with meaningful outputs. This number will either be 0 where the Translated Workflow does not match the original, or 1 where the Translated Workflow matches the original, manual ArcGIS workflow or only have soft workflow errors and can, therefore, be deemed an alternative workflow. The table below (Table 5.9) presents this metric for each competency question.

Table 5.9 Second Precision Metric for Translated Workflow

Competency Question	Precision Metric	Detail
1	1	No error
2	0	Hard error
3	1	No error
4	0	Hard error
5	1	Soft Error
Average	60%	

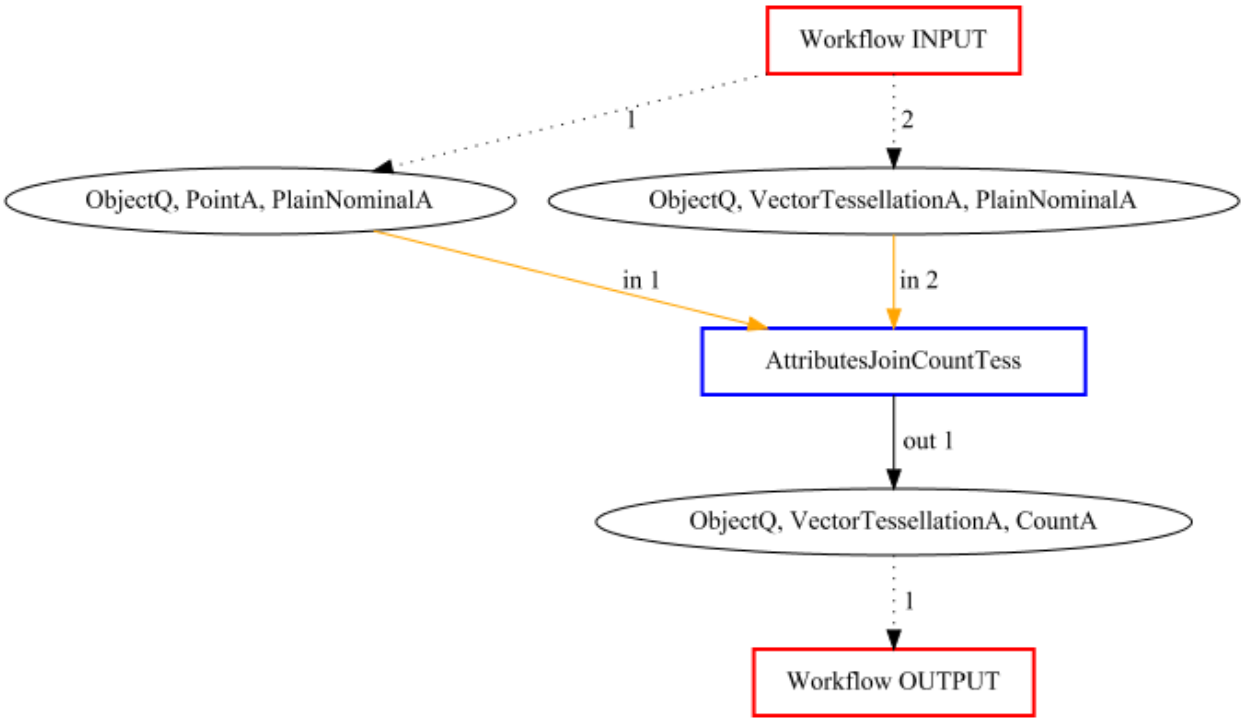
As highlighted by the table above, the translation mechanism is, for the most part, able to predict the correct workflow for a given competency question; proving the ability of the mechanism to successfully translate workflows across workflow environments. It should be noted that the hard workflow error seen in the final workflow for question two is based on the fact that no valid workflows were found in the workflow suggestion set and are likely based on the shortcomings of the synthesis engine used in this mechanism. Considering, however, that these issues with the synthesis engine are still fairly significant, the metric returned here is promising. Additionally, the approval criteria used in this research are very basic and, therefore, do not do much to ensure that invalid workflows are avoided. This is done on purpose to accurately evaluate the translation mechanism without interference from previous assessments. Making these more comprehensive where the translation mechanism is applied in a different setting would mean that completely invalid workflows can be avoided.

5.3 Translated Workflows based on Selection Criteria

Following the evaluation of individual workflow quality and based on the selection criteria for each competency question, the following workflows are presented as the Translated Workflow produced by the translation mechanism. It is these workflows which have been the subject of the second precision calculation discussed above.

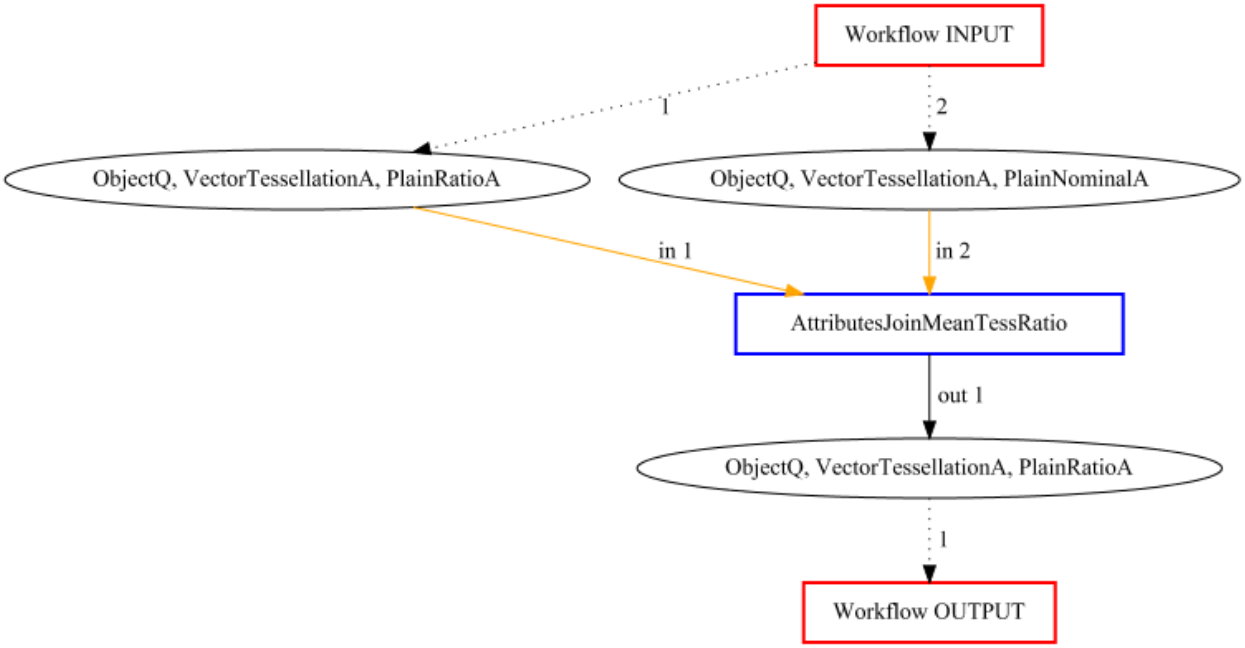
Competency Question 1

Figure 5.7 Translated Workflow from ArcGIS to QGIS for Competency Question 1



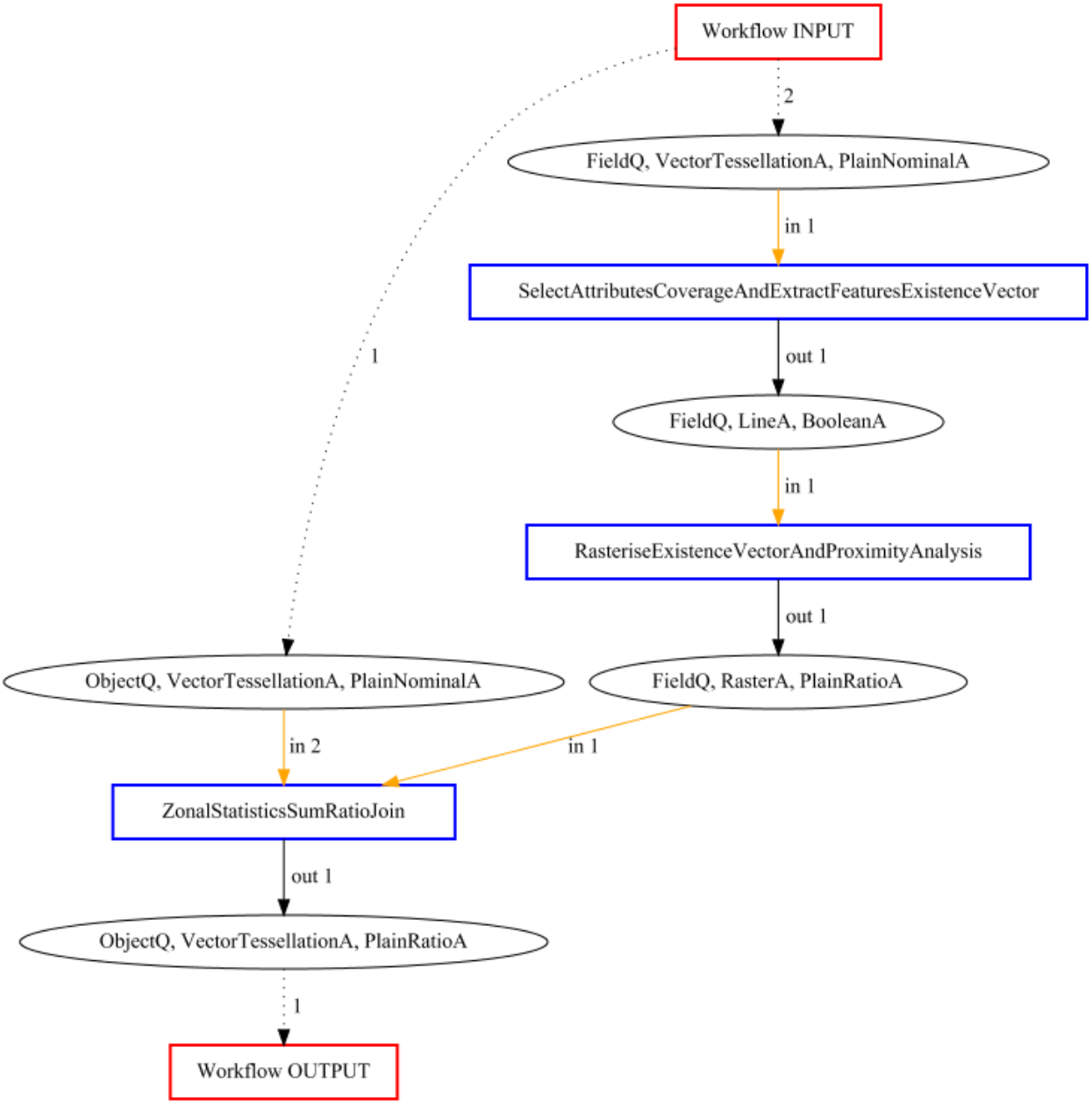
Competency Question 2

Figure 5.8 Translated Workflow from ArcGIS to QGIS for Competency Question 2



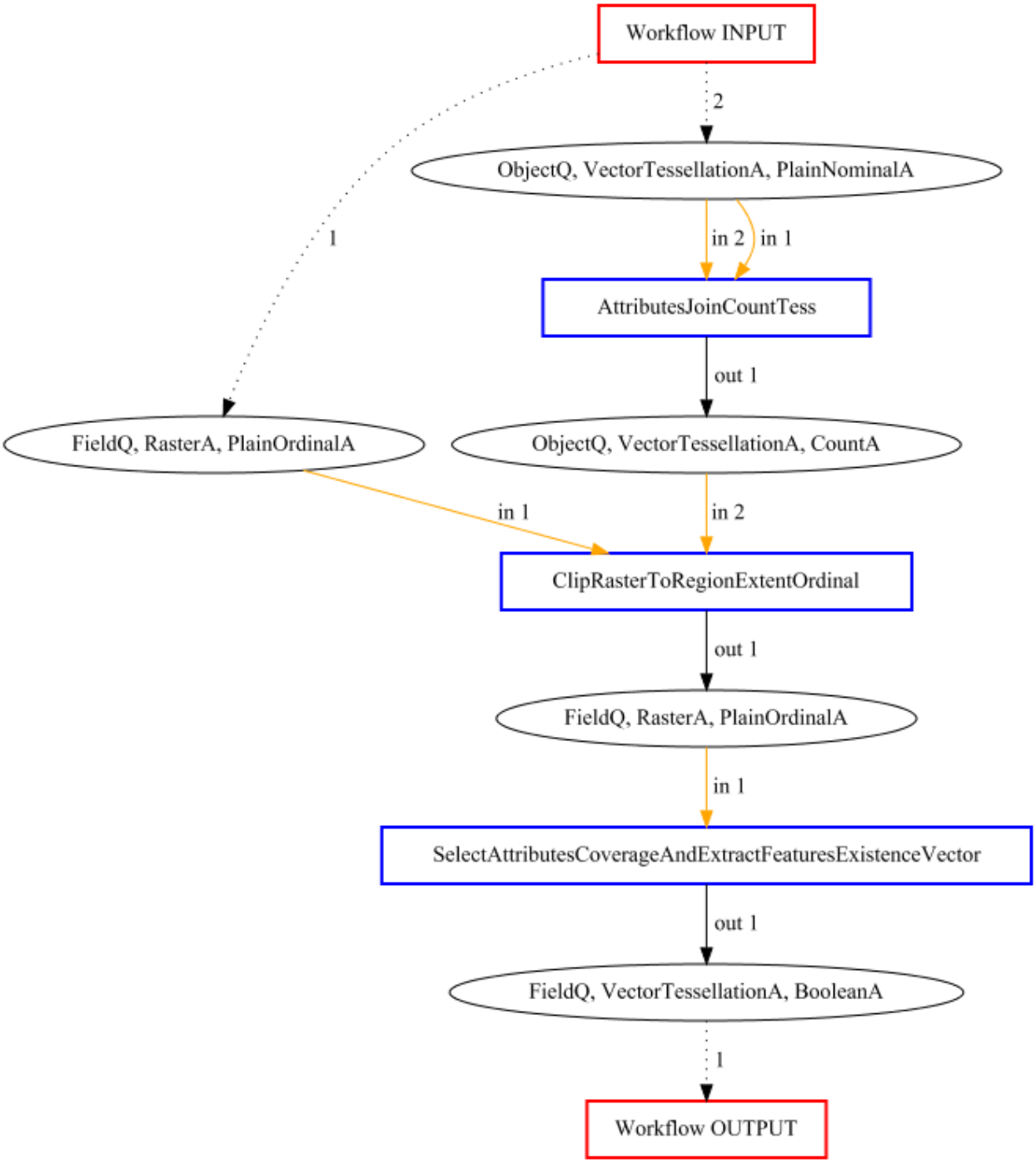
Competency Question 3

Figure 5.9 Translated Workflow from ArcGIS to QGIS for Competency Question 3



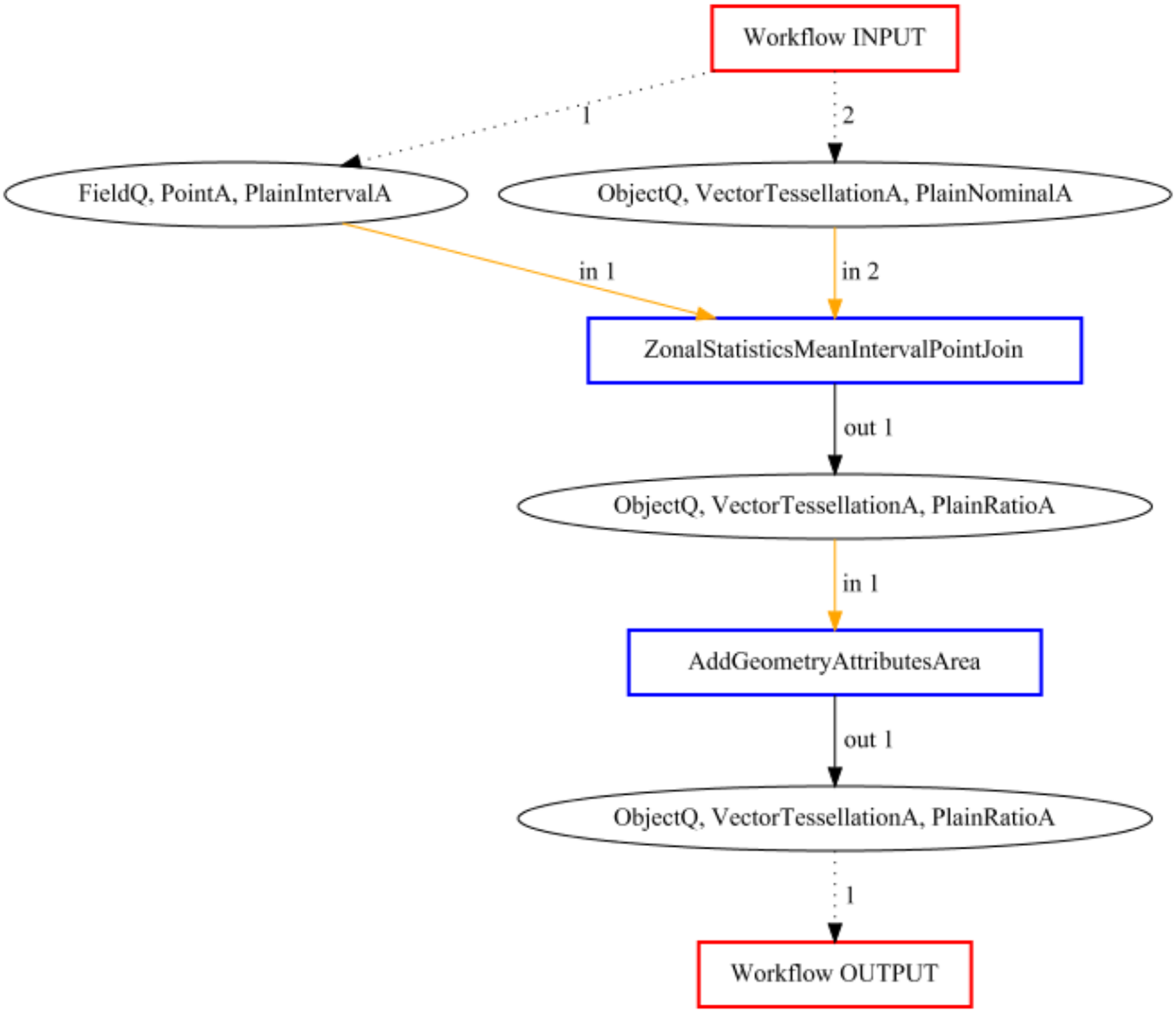
Competency Question 4

Figure 5.10 Translated Workflow from ArcGIS to QGIS for Competency Question 4



Competency Question 5

Figure 5.11 Translated Workflow from ArcGIS to QGIS for Competency Question 5



5.4 Workflow Similarity Assessment

This similarity assessment is closely aligned with that which was formalised by Starlinger et al. (2014) and takes place in four steps. Here, a manually generated ArcGIS workflow will be compared to the set of workflows translated to QGIS workflows. In order to reduce the time intensity of this similarity assessment, particularly because the assessment is conducted in a mostly manual fashion, and because it makes sense to eliminate incorrect workflows, the number of workflows used were limited to only those which did not contain hard workflow errors in the previous evaluation phase. Although this results in different numbers of workflows being included in the similarity assessment per competency question, the assessment will still result in valid, albeit limited in terms of representation, similarity metrics. Indeed, where more workflows can be included in the assessment, the similarity will be more representative of the actual similarity value within a workflow set but because the numbers of correct workflows varied quite considerably from question to question, this approach to the assessment seemed to be the most logical.

Step 1: Pairwise Module Comparison using Ratcliff-Obershelp Similarity

The method of pairwise module comparison used is the Ratcliff-Obershelp Similarity measure. The similarity between workflows is computed by taking three characteristics of a tool (data type inputs, outputs and tool labels) and transforming these into string format for both workflows. These details were gathered from task descriptions for ArcGIS workflow and graphical output for QGIS workflows. For each characteristic, the similarity between the string associated with the manual workflow and that of the translated workflow is computed by dividing the matching number of characters by the total number of characters in both strings combined. This is done 3 times for every tool to calculate similarity of labels, similarity of data input type and similarity of data output type. Please note, where there were two inputs for a specific tool or step in the workflow, the string format for each was simply combined (see below).

The following table illustrates the three-string formats for both the ArcGIS and QGIS workflow for competency question 1 (Table 5.10). Although the manual and the translated workflow happened to be the same length for this particular competency question, this is not always the case. Where manual and translated workflows are different lengths, the table is simply extended to accommodate the additional tools and the corresponding field for the other workflow left blank. This results in a Ratcliff-Obershelp measure of 0 because the blank input results in a lack of similarity between the inputs.

Table 5.10 Specification to String Format Transformation (Competency Question 1)

Tool No.	Characteristic	Manual ArcGIS Workflow	Translated QGIS Workflow
1	Data Input Type	ObjectQPointAPlainNominalA ObjectQVectorTessellationA PlainNominalA	ObjectQ ObjectQVectorTessellationA PlainNominalA PointAPlainNominalA
1	Data Output Type	ObjectQVectorTessellationA CountA	ObjectQVectorTessellationA CountA
1	Tool Label	SpatialJoinCountTess	AttributesJoinCountTess

Following the manual transformation of workflows into their respective string formats, the computation of the Ratcliff-Obershelp similarity measure is calculated for each workflow in the suggestion set based on a Python script which makes use of the DiffLib library (see footnote 6 for Python script). This results in the following table (Table 5.11) for an example QGIS workflow for competency question 4.

Table 5.11 Example of Ratcliff-Obershelp Measure for Competency Question 4

Tool No.	Characteristic	Manual ArcGIS Workflow	Translated QGIS Workflow	Ratcliff-Obershelp Measure
1	Data Input Type	FieldQTessellationAPlain OrdinalA	ObjectQVectorTessellationAPl ainNominalA	0.70
1	Data Output Type	FieldQRasterAPlain OrdinalA	ObjectQVector TessellationAPlainRatioA	0.38

1	Tool Label	ContourToFieldRaster	AddGeometryAttributes Lattice	0.21
			Module Average	**0.43**
2	Data Input Type	FieldQRasterAPlain OrdinalAObjectQVector TessellationAPlain NominalA	FieldQRasterAPlain OrdinalAObjectQVector TessellationAPlainRatioA	0.92
2	Data Output Type	FieldQRasterAPlain OrdinalA	FieldQRasterAPlain OrdinalA	1.00
2	Tool Label	ClipRasterToLatticeExtent Ordinal	ClipRasterToLatticeExtent Ordinal	1.00
			Module Average	**0.97**
3	Data Input Type	FieldQRasterAPlain OrdinalA	FieldQRasterAPlain OrdinalA	1.00
3	Data Output Type	FieldQRasterABooleanA	FieldQRasterABooleanA	1.00
3	Tool Label	CalculatorInequalities NominalOrdinal	CalculatorInequalities NominalOrdinal	1.00
			Module Average	**1.00**

Step 2: Module Mapping

The similarity scores generated in the first step compare the tool modules from the original ArcGIS workflow to those in the QGIS workflow in the order they appear when translated and generated by the synthesis engine. Because the workflows are generated based solely on the task specification, the order that the modules appear in in the generated QGIS workflow does not necessarily mean that these tools are similar or equivalent to those used in the same tool module position in the ArcGIS workflow; only that the order to valid enough to meet the constraints set by the task specification. Indeed, if workflows are differing lengths or included redundant tools, a step by step or tool module by tool module comparison, such as the one done in the previous step, may not be a true reflection of the actual similarity between the translated workflows. To achieve a more meaningful similarity assessment on a workflow by workflow basis, a shuffling of the tool modules as a preliminary step to module mapping and topological comparison is, therefore, required to identify the tool modules between a set of workflows generated for a particular competency question which have the highest similarity to one another.

To achieve this, each ArcGIS tool module from the manual workflow is compared to exactly one QGIS tool module contained within the translated workflow for each shuffling iteration. The shuffling process is repeated several times to ensure that each ArcGIS tool module is compared to each QGIS tool module in a workflow exactly once. As such, if the length of both the ArcGIS workflow and the translated QGIS workflow is 3, the shuffling process needs to be done three times. This shuffling is done to identify the modules between the manual and translated workflows that are the most similar to each other and can, therefore, be preliminarily mapped to one another in this step. The same Python script used in the first step

is also used in this step. For the same workflow above, the following table (Table 5.12) illustrates the results of the similarity comparison.

Table 5.12 Example of Module Mapping and Ratcliff-Obershelp Measure for Competency Question 5

Tool No.	Characteristic	Manual ArcGIS Workflow	Translated QGIS Workflow	Ratcliff-Obershelp Measure
1	Data Input Type	FieldQPointAPlainIntervalAObjectQVectorTessellationAPlainNominalA	FieldQPointAPlainIntervalAObjectQVectorTessellationACountA	0.88
1	Data Output Type	FieldQRasterAPlainIntervalA	ObjectQVectorTessellationAPlainRatioA	0.38
1	Tool Label	ClipPointMeasuresToLatticeExtent	ZonalStatisticsMeanIntervalPointJoin	0.29
			Average	**0.52**
2	Data Input Type	FieldQRasterAPlainIntervalAObjectQVectorTessellationAPlainNominalA	ObjectQVectorTessellationAPlainNominalAObjectQVectorTessellationAPlainNominalA	0.74
2	Data Output Type	ObjectQVectorTessellationAPlainIntervalA	ObjectQVectorTessellationACountA	0.81
2	Tool Label	ZonalStatisticsMeanIntervalJoin	AttributesJoinCountTess	0.33
			Average	**0.63**

Whilst this step certainly provides insight into the tool modules within workflows being compared that are the most similar to each other, the workflows used in this research are DAG's and, therefore, another step in the assessment of similarity is required to deduce actual workflow similarity.

Step 3: Topological Workflow Comparison using Set of Paths Analysis

Because the workflow sets used in this research are examples of direct graphs (DAGs), meaning the modules in a workflow need to appear in a specific order to be valid, a topological comparison which makes use of a set of paths and maximum weight non-crossing measurement also needs to be done before the similarity score for each set of workflows can be returned. All this process of topological comparison amounts to is a remapping of modules in workflow sets which have different lengths to ensure that the modules being compared are modules which are most a like according to expert knowledge. Indeed, where workflows are the same length, any shuffling of modules will produce crossing in mapping modules and, therefore, be eliminated by the process of a set of paths comparison; thus, returning the original translated workflow for similarity comparison. Once the modules across the workflow sets have been mapped to each other using this analysis, the pairwise similarity scores of the mapped modules are then averaged to produce a summary similarity measure.

The following table (Table 5.13) summarises this average similarity score for each competency question. Because the workflows used in some workflow sets are different in lengths, those where there is an uneven

number of tool modules results in the ‘spare’ modules being mapped to a ‘blank node’ which then results in a similarity score of zero. When this is averaged as the sum of the total similarity score per tool module divided by workflow length, the result is already implicitly normalised for workflow length and is, therefore, a global similarity score for the workflow set.

Table 5.13 Average Normalised Similarity Score per Competency Question

Competency Question	No. Of QGIS Workflows	Average Length of QGIS Workflow	Average Normalised Similarity Score
1	12	1.92	0.52
3	5	3.2	0.72
4	17	3	0.81
5	19	2	0.69

From the table above, the similarity between the manually generated ArcGIS workflow and the translated QGIS workflows is at least 50% and in competency question 4, the mechanism was able to translate workflows with a similarity score of 81% based on the tools present in the annotated tool database. The lowest similarity score presented is such because the ArcGIS workflows were different lengths to the QGIS workflows due to tool redundancy and should be read in this context to provide nuance when similarity between workflows for this competency question is assessed. This result highlights the ability of the translation mechanism to both produce quality individual workflows as well as translate workflows with a high degree of similarity across software environments. It should be noted, however, that this score cannot be said to show tool equivalence or lack thereof due to the fact that, arguably, data type inputs, outputs and tool labels do not necessarily translate across environments exactly nor do tools across environments work in exactly the same way. The assessment of tool equivalence across software environments, while worthy of attention, is beyond the scope of this research but the results presented here are a first step towards such an assessment.

Preliminarily, this research does also allow conclusions to be drawn as to the similarity of the software environments used in this research, albeit based only on tool labelling conventions. Indeed, if we assume that QGIS tool modules which have a similar tool label to that in ArcGIS are similar in nature, and if we assume that strict labelling conventions have been followed in this research, which they have, a preliminary conclusion that there is a fairly large degree of similarity between software environments can be made. For more robust conclusions to be made on software environment similarity, however, further research should be done with alternative similarity assessment approaches.

As is clear, this assessment procedure produces the conclusion that the workflows generated by the translation mechanism do produce workflows that are largely similar to each other based on their tool annotations and tool labels. There are, however, matters worth mentioning following the presentation of this result. Firstly, the quality labels used for the similarity comparison are largely dependent on the labelling conventions used by the expert when developing the annotated tool databases. Although the labelling conventions used in this research were closely related to the actual tool names used in the software environments and the data type inputs required for sub-tooling, where labelling conventions are not strictly

followed with reference to the software environments that the tools are being taken from, this has the potential to skew similarity scores both positively and negatively. As such, care should be taken when reproducing this research in other environments. Secondly, more complicated workflows and their translations may require more robust, complex, and potentially automated graph comparison techniques. Again, while the techniques used in this research were more than sufficient for the context, this should be taken account of when reproducing the research.

Chapter 6 Conclusion and Future Research

As an extension of the growing body of literature on the use of synthesis engines to automate workflow synthesis to aid in the use, sharing, reuse and adaptation of already existing GIS workflows in different contexts, the research presented above aimed to develop a mechanism to translate these workflow across software environments and tested the robustness of this mechanism by evaluating the generated workflows for quality and similarity. The wider application of this translation mechanism is to assist with the ability of analysts and non-experts alike to implement GIS workflows beyond the confines of current software environments. The point on the horizon in concluding this research is that tools are able to be sufficiently abstracted beyond their software environments and contained within a repository wherein tool comparisons based on similarity across environments are available to guide the choice of tools and software used in the eventual analysis. The translation mechanism highlights the potential to also synthesis workflows from this abstracted tool repository, removing the knowledge barrier currently present in the field, improving the accessibility of the technology and the sharing of its workflows across contexts.

To guide this research project, several research questions were posed. Firstly, the question investigating what the technical inclusions for the automatic translation of workflows across software environments were led to the identification of a relevant ontology, the CCDT ontology, and the APE synthesis engine for this purpose. Here, annotated tool databases for ArcGIS tools and QGIS tools based on the CCDT ontology were developed and competency questions used to generate task specifications on which the APE engine configured according to these specifications. The identification of these technical components created the formalised translation mechanism presented. Secondly, the investigation of the capacity of the CCDT ontology for describing tools across software environments and its ability to support of the translation mechanism in producing quality outputs led to the implementation of the error-based method of individual workflow quality evaluation. The results of this investigation highlighted both the ability of the translation mechanism to produce quality workflows in both software environments as well as the capacity of the CCDT ontology to successfully abstract software-specific tools well enough to allow for workflow translation with quality and valid outputs. Finally, the investigation into the similarity of the workflows translated from ArcGIS to QGIS was achieved through the implementation of Starlinger et al.'s (2014) approach to structure-based workflow comparison for similarity. Here, the similarity between sets of workflows translated across environments were no less than 50%, speaking to the capabilities of the translation mechanism used in this research to reproduce domain knowledge across GIS software environments using these technical components do a degree of success.

While the results of the above investigation are promising and do show a potential for the translation mechanism to be used in further research and as part of aided workflow synthesis mechanisms, the results of the evaluations and assessments as well as the nuances of the research do highlight that this mechanism is not as robust as it could be. Indeed, for the mechanism to fully reproduce domain knowledge, it is likely that the ontology used in this research will need to be expanded to formalise more sub-operations or be replaced with an ontology which is better able to do so. The synthesis engine, its configurations and constraints may also need to be adapted to support the translation of better-quality workflows. These issues and suggestions are discussed at length in the following section. Overall, the results produced by this research, particularly the quality levels and similarity scores, do present promising evidence that this

mechanism can support workflow translation and accurate reproduction of domain knowledge. Further development of this mechanism is, however, needed for this goal to be achieved.

6.1 Future Research

Due to the time limitations of this research thesis, there are several issues which are worthy of addressing in the context of this research which were simply out of scope. As such, these will be presented here as a guide to how this research can be improved and built upon in the future. These are presented in the sub-sections below.

6.1.1 Annotated Tool Databases

Because of the time limitations set of this research project, the tools included in the annotated tool databases used in this research were largely limited to those which were required to synthesis workflows in both environments. Although the ArcGIS annotated tool database was larger because previous research had formalised some tools in these databases previously, these tools only represent a small fraction of those in each environment and, indeed, even where tools were annotated, not all sub-tools were formalised. The limited nature of the databases presents two areas of future research. Firstly, the current databases which make use of the CCDT ontology do not accurately formalise tools were distance, density or statistical methods are implemented and, therefore, limit the capacity for the translation mechanism to answer a wide range of spatial questions. Although this speaks too of the capacity of the ontology, expanding the tool databases to include tools which are properly specified for these operations would be a step towards a better formalisation of the knowledge domain relating to workflow synthesis and translation. Secondly, because the tool database was so limited, the quality of the workflows translated when the databases are much larger is not assessed. Preliminary interactions with a larger database as part of this research indicate that where to database is larger and only annotated using the CCDT ontology, the mechanism and APE engine may produce more errors in generating workflows. This should be investigated.

6.1.2 Further Workflow Constraints and the Reduction of Other Constraints

Firstly, the discussions presented in previous chapters of this thesis acknowledge the possibility of constraining workflows further than was done in this research when configuring the APE engine. It is probable that constraining workflows based on the forced appearance of certain tools or data sequences may improve the quality of the workflows produced. Indeed, the further specification of data types in the tool annotations has been shown to improve overall workflow quality as highlighted by Kruiger et al. (2020). Again, this should be investigated in future research. Secondly, and indeed conversely, the constraint that specifies workflows should be generated in terms of a maximum number in a set rather than all workflows with length up to a certain figure means that the engine essentially produces the first 20 workflows is computes and generally in shortest length order. The workflows generated above are likely not the full set of workflows of a particular length and, therefore, may exclude more relevant workflows from the set of translated workflows. For future work, these changes to constraints and configuration could increase the quality and validity of the workflow set to a particular spatial question.

6.1.2 Software Environment Similarity

As mentioned briefly above, the similarity assessment done as part of this research does preliminary suggest similarities between the QGIS and ArcGIS software environments. Although it became apparent that another similarity assessment was beyond the scope of this research due to time limitations, a further similarity assessment to more robustly compute software environment similarity should be explored and can be supported by Bergmann and Gil (2012)'s framework. The goal of such a similarity assessment would be to establish some level of tool equivalence between GIS tools from different environments and is based on an annotation-based strategy. This is argued to be the most appropriate for this sort of similarity for a number of reasons. Firstly, although structure-based approaches such as those used above may perform better than annotation-based assessments for assessing similarity between individual workflows, it can be argued that true functional similarity between workflows as well as between tools and between different software environments in this case can only be assessed through the use of semantics designed to capture these functional similarities. The prevalence of these semantics in this research means that annotation-based similarity assessment is possible and, indeed, important for this research. Indeed, in the case of this research, the use of the CCDT ontology is argued to abstract the workflows beyond software environment-specific details to capture the true functional similarities between tools included in the workflows. Assessment of similarity between the semantics used to capture these can therefore support conclusions made on similarities between software environments. Secondly, it should be noted that data types annotations are not the only semantics included in the semantic annotations. GIS tools are also labelled (rdfs:label) with their tool names and sub-operation names closely related to their parameters as well as their descriptions. These annotations may allow for semantic similarity assessments to also assess some level of naming convention or implementation similarity which may further support a conclusion on environment similarity. This approach to similarity assessment of software environments should be explored in future research related to workflow translation.

References

Albrecht, J., Derman, B., & Ramasubramanian, L. (2008). Geo-Ontology Tools: The Missing Link. *Transactions in GIS*, 12(4), 409-424.

Barga, R., & Gannon, D. (2007). Scientific versus Business Workflows. In *Workflows for e-Science*, (pp. 9-16). London, United Kingdom: Springer.

Bergmann, R., & Gil, Y. (2014). Similarity Assessment and Efficient Retrieval of Semantic Workflows. *Information Systems*, 115-127.

Berners-Lee, T., & Hendler, J. (2001). The Semantic Web. *Scientific American*, 284(5), 34–43.

Bernstein, M. S.; Little, G.; Miller, R. C.; Hartmann, B.; Ackerman, M. S.; Karger, D. R.; Crowell, D.; and Panovich, K. (2010). Soylent: A Word Processor with a Crowd Inside. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology* (pp. 313-322). UIST '10:ACM.

Bisht, M. (n.d). Measures of Semantic Similarity. Retrieved from: https://www.3pillarglobal.com/insights/measures-of-semantic-similarity.

Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3), 1–22.

Brauner, J. (2015). Formalisation for Geooperators-Geoprocessing in Spatial Data Infrastructures. Ph.D Thesis, Technische Universitat Dresden.

Chun, S.A., Atluri, V., Adam, N.R. (2002). Domain knowledge-based automatic workflow generation. 81-93. Berlin, Germany: Springer.

Clarke, K.C. (1986). Advances in Geographic Information Systems. *Computers, Environment and Urban Systems*, 10(3-4), 175-184.

Clempner, J. B. (2017). Classical workflow nets and workflow nets with reset arcs: using Lyapunov stability for soundness verification. *Journal of Experimental & Theoretical Artificial Intelligence*, 29(1), 43–57.

Chrisman, N. (2002). Chapter: Reference Systems for Measurement. In *Exploring Geographic Information Systems*, 2nd Edition (pp. 15-35). Wiley.

Costa, F., de Oliveira, D., Ogasawara, E., Lima, A. A., and Mattoso, M. (2012). Athena: text mining-based discovery of scientific workflows in disperse repositories. In *Resource Discovery*, (pp. 104–121). Springer.

Deelman, E., Gannon, D., Shields, M. & Taylor, I. (2009). Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25(5), 528-540.

Deelman, E., Singh, G., Su, M.-H., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Vahi, K.,

Berriman, G.B., Good, J., Laity, A., Jacob, J., Katz, D. (2005). Pegasus: A framework for mapping

complex scientific workflows onto distributed systems. *Scientific Programming*, 13(3), 219–237.

Donnelly, M., & Guizzardi, G. (Eds.). (2012). Formal Ontology in Information Systems. In *Proceedings of the Seventh International Conference FOIS 2012.* IOS Press.

Fahringer, T., Prodan, R., Duan, R., Nerieri, F., Podlipnig, S., Qin, J., Siddiqui, M., Truong,

H., Villazon, A., Wieczorek, M. (2005). ASKALON: A grid application development and computing environment. In *Proceedings from the 6th IEEE/ACM International Workshop on Grid Computing.*

Fitzner, D., Hoffmann, J., & Klien, E. (2011). Functional description of geoprocessing services as conjunctive data log queries. *GeoInformatica*, 15(1), 191–221.

Fox, G.C., Gannon, D. (eds.) (2006). Concurrency and Computation: Practice and Experience. *Special Issue: Workflow in Grid Systems*, 18(10). Chichester: John Wiley & Sons.

Freitag, B., Steffen, B., Margaria, T., & Zukowski, U. (1995). An Approach to Intelligent Software Library Management. In Proceedings of the 4th International Conference on Database Systems for Advanced Applications. (pp. 10-13).

Friesen, N. & Ruping, S. (2010). Workflow Analysis Using Graph Kernels. SoKD.

Gangemi, A., Catenacci, C., Ciaramita, M., & Lehmann, J. (2005). A theoretical framework for ontology evaluation and validation. Laboratory of Applied Ontology. Rome.

Gangemi, A. (2005). Ontology Design Patterns for Semantic Web Content, Musen et al. (eds.) *Proceedings of the Fourth International Semantic Web Conference*, 262-276. Retrieved from https://link.springer.com/chapter/10.1007/11574620_21

Garijo, D., Alper, P., Belhaijame, K., Corcho, O., Gil, Y & Goble, C. (2014). Common motifs in scientific workflows: an empirical analysis. *Future Generation Computer Systems*, 36, 338-351.

Gensesereth, D.A & Nilsson, N.R. (1987). *Logical Foundation of Artificial Intelligence* (2nd edition). Los Atlos, United States: Morgan and Kaufmann.

Gil, Y. (2007). Workflow Composition: Semantic Representations for Flexible Automation. In *Workflows for e-Science* (pp. 244–257). London, United Kingdom: Springer London.

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox G., Gannon, D., … Myers, J. (2007). Examining the challenges of scientific workflows. *Computer,* 40(12), 24-32.

Goderis, A., Sattler, U., Lord, P., & Goble, C. (2005). Seven bottlenecks to workflow reuse and repurposing. In *Proceedings from the Internal Semantic Web Conference (pp.* 323-337). Galway: Springer Berlin.

Goderis, A., Li, R and Goble, C. (2006). Workflow discovery: the problem, a case study from e-science and a graph-based solution. In International Conference on Web Services (ICWS) 2006, pages 312–319. IEEE.

Gomez-Perez, A. (2001). Evaluation of Ontologies. International Journal of Intelligent Systems. 16(3).

Gomez-Perez, A., Fernández-López, M., & Corcho, O. (2006). *Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer Science & Business Media.

Gruber, T.R. (1995). Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5-6). 907-928.

Guarino, T.R (1997). Understanding, building, and using ontologies. *International Journal of Human-Computer Studies*, 46(2-3), 293-310.

Guarino, N. (1998). Formal ontology in information systems. In *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (FOIS 98)*. Amsterdam: IOS Press.

Gomez-Perez, A. (2001). Evaluation of Ontologies. *International Journal of Intelligent Systems*, 16(3), 391-409.

Hitzler, P., Gangemi, A., & Janowicz, K. (2016). *Ontology Engineering with Ontology Design Patterns: Foundations and Applications.* Amsterdam: IOS Press/Ohmsha.

Hofer, B., Mäs, S., Brauner, J., & Bernard, L. (2017). Towards a knowledge base to support geoprocessing workflow development. *International Journal of Geographical Information Science*, 31(4), 694–716.

Hwang, E., Glass, A.P, Gutzmann, J. and Shin, K. J. (2008). The Meaning of a Liveable Community for Older Adults in the United States and Korea. *Journal of Housing for the Elderly*, 22(3).

Kasalica, V., & Lamprecht, A. L. (2018). Automated composition of scientific workflows: A case study on geographic data manipulation. In *Proceedings from the 2018 IEEE 14th International Conference on e-Science* (pp. 362-363). IEEE.

Kasalica V., Lamprecht A. L. (2019) Workflow Discovery Through Semantic Constraints: A Geovisualization Case Study. In Misra S. et al. (eds), *Computational Science and Its Applications – ICCSA 2019.* ICCSA 2019: Springer, Cham.

- (2020a) APE: A Command Line Tool and API for Automated Workflow Composition. Under Review.
- (2020b) APE: Workflow Discover with Semantic Constraints: A SAT-Based Implementation. To appear in ECEASST.

Kent, A., Berry, M. M., Luehrs, F. U. Jr, Perry, J. W. (1955) Machine Literature Searching VIII: Operational criteria for designing information retrieval systems. *American Documentation* 6, 93–101.

Kittur, A., Smus, B., Kraut, R., and Khamkar, S. (2011). Crowdforge: Crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology UIST '11.*

Koohi-Var, T., & Zahedi, M. (2018). Cross-domain similarity assessment for workflow improvement to handle Big Data challenge in workflow management. *Journal of Big Data*, 26. https://doi.org/10.1186/s40537-018-0135-6

Kruiger, H., Meerlo, R, Kasalica, V., Lamprecht, A.L., Scheider, S. (2020). Loose programming of GIS workflows with geo-analytical concepts. Unpublished manuscript, Utrecht University, Utrecht, The Netherlands.

Kuhn, W. (2012). Core concepts of spatial information for transdisciplinary research. International *Journal of Geographical Information Science*, 26(12), 2267–2276.

Lamprecht, A. L., Margaria, T., & Steffen, B. (2009). Bio-JETI: A framework for semantics-based service composition. *BMC Bioinformatics*, 10 (Supplement 10).

Lamprecht, A.L., Naujokat, S., Margaria, T. & Steffen, B. (2010). Synthesis-Based Loose Programming. In *2010 Seventh International Conference on the Quality of Information and Communications Technology* (pp. 262-267). IEEE.

Lemmens, R., Wytzisk, A., By, R. d., Granell, C., Gould, M., & van Oosterom, P. (2006). Integrating Semantic and Syntactic Descriptions to Chain Geographic Services. *IEEE Internet Computing*, 10(5), 42–52.

Leymann, F. & Roller, D. (1997). Workflow-based applications. *IBM Systems Journal.* 36(1), 102-123.

Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Lin, C., Lu, S., Lai, Z., Chebotko, A., Fei, X., Hua, J., and Fotouhi, F. (2008). Service-orientated Architecture for VIEW: A Visual Scientific Workflow Management System. In *2008 IEEE International Conference on Services Computing* (pp. 335-342). IEEE.

Ludaescher, B., Altintas, I., Bowers, S., Cummings, J., Critchlow, T., Deelman, E., Freire, J., Roure, D.D., Goble, C., Jones, M., Klasky, S., Podhorszki, N., Silva, C., Taylor, I., Vouk, M. (2009) Scientific Process Automation and Workflow Management. In Shoshani, A., Rotem, D. (eds.), Scientific Data Management: Challenges, Existing Technology, and Deployment. Chapman and Hall/CRC.

Ludaescher, B., Weske, M., McPhillips, T., & Bowers, S. (2009). Scientific workflows: Business as usual? In *Proceedings from Business Process Management 7th International Conference BPM 2009* (pp. 31-47). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). DOI: https://doi.org/10.1007/978-3-642-03848-8_4.

Lutz, M. (2007). Ontology-Based Descriptions for Semantic Discovery and Composition of Geoprocessing Services. *GeoInformatica*, 11(1), 1–36.

Maheshwari, K. and Montagnat, J. (2010). Scientific Workflow Development Using Both Visual and Script-Based Representations. In *Proceedings from 2010 6th World Congress on Services*.

Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuiness, D., … Sycara, K. (2005). Bringing semantics to web services: the OWL-S approach. In *proceedings from International Workshop on Semantic Web Services and Web Process Composition* (pp. 26-42). Lecture Notes in Computer Science (LNCS, volume 3387). Berlin, Germany: Springer.

Mayank, M. (2019, February 3). String Similarity – The Basic Know Your Algorithms Guide! Retrieved from: https://itnext.io/string-similarity-the-basic-know-your-algorithms-guide-3de3d7346227

Meerlo, R. (2019). The Relevance of Semantics for GIS Workflow Synthesis. Master's Thesis. Utrecht University, The Netherlands. URL: http://geographicknowledge.de/pdf/Thesis%20Rogier%2016-08-2019.pdf

Munir, K., & Sheraz Anjum, M. (2018). The use of ontologies for effective knowledge modelling and information retrieval. *Applied Computing and Informatics*, 14(2), 116-126.

Noronha, J.; Hysen, E.; Zhang, H.; and Gajos, K. Z. (2011). Platemate: Crowdsourcing nutrition analysis from food photographs. In *Proceedings of the 24th annual ACM symposium on User interface software and technology* (pp. 1-12).

Oliveira, W., de Oliveira, D & Braganholo, V. (2015). *Experiencing PROV-Wf for provenance interoperability in SWfMSs*. Cham: Springer.

QGIS. (n.d). Vector Overlay – Union. Retrieved from https://docs.qgis.org/3.10/en/docs/user_manual/processing_algs/qgis/vectoroverlay.html#union.

Ratcliff, J. W. & Metzener, D. (1988). Pattern Matching: The Gestalt Approach. *Dr Dobb's Journal*.

Raad, J., & Cruz, C. (2015). A Survey on Ontology Evaluation Methods. In Proceedings of the *International Conference on Knowledge Engineering and Ontology Development*. Part of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. Lisbon, Portugal.

Ruth, M. and Franklin, R.S. (2014). Liveability for All? Conceptual Limits and Practical Implications. *Applied Geography*, 49, 18-23.

Santos, E., Lins, L., Ahrens, J. P., Freire, J., and Silva, C. T. (2008). A first study on clustering collection of workflow graphs. In Provenance and Annotation of Data Processes (pp. 160-173). Springer.

Scheider, S. (2019). Semantic data type signatures for representing spatial core concepts in GIS operation on spatial layers. Utrecht: Utrecht University.

Scheider, S. and Ballatore, A. (2018). Semantic typing of linked geoprocessing workflows. *International Journal of Digital Earth*, 11(1), 113-138.

Scheider, S. and Tomko, M. (2016). Knowing whether Spatio-temporal analysis procedures are applicable to datasets. Paper presented at the *9th International Conference on Formal Ontology in Information Systems (FOIS 2016)*. Annecy, France.

Scheider, S., Meerlo, R., Kasalica, V. and Lamprecht, A.-L. (2020) Ontology of core concept data types for answering geoanalytical questions. Journal of Spatial Information Theory. URL: https://www.josis.org/index.php/josis/article/view/555. Under review.

Scheider, S., Ostermann, F & Adams, B. (2016). Why good data analysts need to be critical synthesists. Determining the role of semantics in data analysts. *Future Generation Computer Systems*, 72, 11-12.

Silva, V., Chirigati, F., Maia, K., Ogasawara, E., Oliveira, D., Braganholo, V., Murta, L and Mattoso, M. (2011). Similarity based workflow clustering. *Journal of Computational Interdisciplinary Sciences*, 2(1):23-35.

Slimani, T. (2013). Description and Evaluation of Semantic Similarity Measures Approaches.

Starlinger, J. (2015). Similarity Measures for Scientific Workflows. Doctoral Thesis. Humbolt University, Berlin, Germany. URL: https://edoc.hu-berlin.de/bitstream/handle/18452/18058/starlinger.pdf?sequence=1

Starlinger, J., Brancotte, B., Cohen-Boulakia, S. and Leser, U. (2015). Similarity Search for Scientific Workflows.

Starlinger J., Cohen-Boulakia, S., Khanna, S., Davidson, S. B., and Leser, U. (2014). Layer Decomposition: An Effective Structure-based Approach for Scientific Workflow Similarity. In *2014 IEEE 10 International Conference on e-Science.* IEEE.

Steffen, B., Margaria, T., & Freitag, B. (1993). *Module Configuration by Minimal Model Configuration*. Passau.

Stehman, Stephen V. (1997). "Selecting and interpreting measures of thematic classification accuracy". Remote Sensing of Environment. 62 (1): 77–89.

Stoyanovich, J., Taskar, B., and Davidson, S. (2010). Exploring repositories of scientific workflows. *WANDS*.

Son, J. H., Sun Kim, J., & Ho Kim, M. (2005). Extracting the workflow critical path from the extended well-formed workflow schema. *Journal of Computer and System Sciences*, 70(1), 86-106.

Taniar, D. (Ed.). (2006). *Web Semantics & Ontology*. IGI Global.

Ubels, S. (2018). Understanding abstract geo-information workflows and converting them to executable workflows using semantic web technologies. Master's Thesis, The University of Twente.

Van der Aalst, W.M.P. (1998). The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems and Computers* 8(1):21-66.

Van der Aalst, W.M.P. and Lassen, K.B. (2008). Translating unstructured workflow processes to readable BPEL: Theory and Implementation. *Journal of Information and Software Technology,* 50(3), 131-159.

Vouk, M.A., Singh, M.P. (1996). *Quality of Service and Scientific Workflows*. North Carolina State University at Raleigh.

Wilson, J. M. (2003). Gantt Charts: A Centenary Appreciation. *European Journal of Operational Research*, 149(2), 430-437.

Workflow [Def 1]. (n.d) In Merriam-Webster Online, retrieved November 11, 2019, from https://www.merriam-webster.com/dictionary/workflow

Xiang, X. & Madey, G. (2007) Improving the Reuse of Scientific Workflows and Their By-Products. ICWS, 792-799.

Yue, P., Di, L., Yang, W., Yu, G., & Zhao, P. (2007). Semantics-based automatic composition of geospatial Web service chains. *Computers & Geosciences*, 33(5), 649–665.

Zhang, H., Horvitz, E., & Parkes, D.C. (2013). Automated Workflow Synthesis. In AAAI Conference on Artificial Intelligence., North America.

Zhang, J., Tan, W., Alexander, J., Foster, I. & Madduri, R. (2011). Recommend-As-You-Go: A Novel Approach Support Services-Oriented Scientific Workflow Reuse.

Zuniga, G. L. (2001). Ontology: its transformation from philosophy to information systems. In *Proceedings of the International Conference on Formal Ontology in Information Systems*. (pp 187-197). New York, New York, USA: ACM Press.

Appendix A Drive Links

1. GitHub Repository for APE Configuration and Workflow Outputs: https://github.com/lexirowland/workflowtranslation
2. Python Script for Similarity Assessment: https://github.com/lexirowland/workflowtranslation/blob/master/similarity_assessment/ratcliff_obershelp_measure_automated.py
3. Google Drive for Quality and Similarity Assessments: https://docs.google.com/spreadsheets/d/1sTZCz4AVK6ZQdZo5wiXff2BC7jmeAMLTpHRE-RmDEEA/edit?usp=sharing