SE in ES: opportunities for software engineering and cloud computing in environmental science

SIMM, W.A., SAMREEN, Faiza <http://orcid.org/0000-0002-9522-0713>, BASSETT, R., FERRARIO, M.A., BLAIR, G., WHITTLE, J. and YOUNG, P.J.

Published version

SIMM, W.A., SAMREEN, Faiza, BASSETT, R., FERRARIO, M.A., BLAIR, G., WHITTLE, J. and YOUNG, P.J. (2018). SE in ES: opportunities for software engineering and cloud computing in environmental science. In: ICSE-SEIS '18: Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Society. ACM Press, 61-70.

SE in ES: Opportunities for Software Engineering and Cloud Computing in Environmental Science

W.A. Simm[1], F. Samreen[1], R. Bassett[2], M.A. Ferrario[1], G. Blair[1], J. Whittle[3] and P.J. Young[2]
[1]School of Computing and Communications, Lancaster University, Lancaster, UK.
[2]Lancaster Environment Centre, Lancaster University, Lancaster, UK.
[3]Faculty of IT, Monash University, Melbourne, Australia.
{w.a.simm, f.samreen, r.bassett, g.blair, m.a.ferrario, paul.j.young}@lancaster.ac.uk, jon.whittle@monash.edu

ABSTRACT

New and emergent computing architectures and software engineering practices provide an opportunity for environmental models to be deployed more efficiently and democratically. In this paper we aim to capture the software engineering practices of environmental scientists, highlight opportunities for software engineering and work towards developing a domain specific language for the configuration and deployment of environmental models. We hold a series of interviews with environmental scientists involved in developing and deploying computer based environmental models about the approach taken in engineering models, and describe a case study in deploying an environmental model (WRF: Weather Research Forecasting) on a cloud architecture. From these studies we find a number of opportunities for A) software engineering methods and tools such as Domain Specific Languages to play a role in abstracting from underlying computing complexity, and for B) new architectures to increase efficiency and availability of deployment. Together, we propose they will allow scientists to concentrate on fundamental science rather than specifics of the underlying computing.

CCS CONCEPTS

• **Applied computing → Environmental sciences**; • **Software and its engineering → Abstraction, modeling and modularity**;

KEYWORDS

Environmental Science, Model Driven Engineering, Cloud Computing, Environmental Modelling, WRF

ACM Reference Format:
W.A. Simm[1], F. Samreen[1], R. Bassett[2], M.A. Ferrario[1], G. Blair[1], J. Whittle[3] and P.J. Young[2]. 2018. SE in ES: Opportunities for Software Engineering and Cloud Computing in Environmental Science. In *Proceedings of ICSE-SEIS'18.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3183428.3183430

1 INTRODUCTION

A key task in Environmental Science (ES) is developing computational models of environmental phenomena so that behaviours can be understood, projected in to the future, and the effect of disturbances can be studied. Environmental modelling is used to evaluate uncertainty, risk assessment, and mitigation strategies around flood/ drought, food security and the impact of climate change (that has major consequences for the economy and society).

Modelling involves working with complex data sets and computing systems with sufficient resources to derive model output in a timely manner. The models can become highly complex tied to specific computing architectures and frequently evolve with both the scientist's understanding of the particular phenomena and the understanding of software and computing architectures. This limits the potential of the models produced, by making them difficult to deploy to alternative architectures, and to interface to other models and systems.

In our work we advocate a novel approach based on a combination of model-driven engineering coupled with software frameworks, which we envisage will enable a paradigm shift in the flexible and tailored support offered by cloud computing for given application domains. We plan to develop Domain Specific Languages (DSLs) and software frameworks to abstract away from complex underlying computing architectures and data processing, developing tools to allow the environmental scientist to concentrate on the science of the environment, and enable the environmental models produced to run on appropriate architectures and interface to other models and systems. We will specifically deploy environmental models, which are traditionally run locally or on High Performance Computing (HPC) facilities, on cloud architectures to take advantage of on demand scaleablity, accessibility and connected cloud services such as data stores and analytics.

In this paper we i) aim to capture the software engineering practices and associated systems administration skills of a diverse group of environmental scientists, from this ii) highlight opportunities for software engineering in this domain and iii) develop a case study deployment of a complex weather model where we create an automated model deployment on cloud architecture. The goal of this work is to understand the domain with a view of influencing future development of a DSL for the configuration and deployment of environmental models in the cloud.

We take a highly user engaged and agile approach, first taking a broad perspective and holding a series of interview / observations with a range of environmental scientists, then gaining focus with feedback on our development with a group of expert users, and will continue development with an embedded group of end users.

This paper makes three main contributions:

ICSE-SEIS'18, May 27-June 3, 2018, Gothenburg, Sweden,

ACM ISBN 978-1-4503-5661-9/18/05.
https://doi.org/10.1145/3183428.3183430

- Interview based study of software engineering practices amongst environmental modellers, highlighting opportunities for SE tools in environmental modelling and opportunities for the SE community in ES.
- Deployment of the WRF model in a cloud architecture with an exploration of the qualities including cost / performance comparison, and feedback from expert users about the possibilities this opens up such as reduction in barriers to use and likely use cases in their work.
- Emerging insight to be used in developing a DSL for deploying environmental models in the cloud.

1.1 Approach

We conduct this research in an agile, highly user-engaged manner taking inspiration and guidance from the Speedplay innovation management framework, which was developed across multiple participatory technology innovation projects [5]. This involves working in multiple iterative "sprints", each a cycle which furthers both domain understanding in users and researcher, and the technology being developed. In this the development team can start building without being experts in the complex domain, rather they are guided by embedded experts as co-researchers and the evaluation of prototypes reveal understanding as the project plays out.

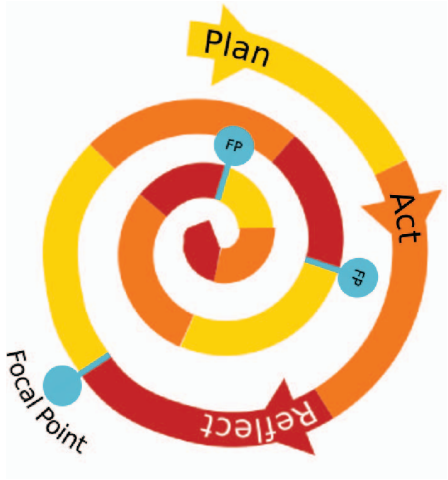

Figure 1: Project Cycles, inspired by Ferrario et al.[5].

Each Sprint is an iterative "plan, act, reflect" cycle which starts with a broad perspective on the chosen domain, and iteratively builds on the last to focus through developing technology and understanding, and ends with a focal point (FP). The spiral in figure 1 represents the research team starting with a broad perspective on the domain, here we engaged a cross section of Environmental Scientists developing or using models in their work, from soil chemistry to bee populations to world climates. These informal interviews both up-skilled the research team with domain knowledge, and an understanding of the many ways in which environmental scientists are developing and deploying models - and the associated SE skills and system administrations tasks.

At the end of the first cycle, the focal point brought the team together to reflect on understanding, review the skills and research interests in the team and choose an avenue of enquiry from the many possibilities. Here the spiral starts to focus, however other avenues (or offshoots) are recorded and may be explored later. At this focal point the team elected to use the Weather Research and Forecasting (WRF) model (see later section) as a case study and developed the technology towards the project goals. The second FP, was a "show and tell" in which a panel of experts who use and develop WRF in their work were shown the developments, and this allowed the team to further focus investigation by brainstorming use cases for the development in their work.

The planned third cycle will engage specific user groups for the selected use cases and we plan to continue this spiral of cycles, each iterating on developments and knowledge gained through the previous. This will lead to a technology that will be a good fit to the intended use, and embeds users and experts in the process frequently for evaluation and feedback and creates engaged domain "champions" of the technology developed.

2 BACKGROUND AND RELATED WORK

2.1 Environmental Modelling

Creating and executing computational models of environmental phenomena is a key task for many environmental scientists. Environmental modelling is a large and diverse research field which spans domains from climate prediction to soil chemistry, and covers global to local scales. Environmental models are typically used to bridge observational paucity and predict the future. This has societal benefits though informed decision making on environmental impacts, for example with respect to climate change and its impacts on extreme events, biodiversity loss and food security.

For a given area, a plethora of environmental models exist each with different assumptions, levels of physical parameterisations, modelling approaches and complexity. Simple environmental models can run on individual workstations while other, more complex, models require dedicated HPCs. Model simulations are often combined as ensembles requiring large computational resources, particularly for climate prediction through: i) running the same model multiple times while varying the starting point (initial condition ensembles [7]) or assumptions (perturbed physics ensembles [11]); or ii) running multiple (distinct) models, created at separate research institutions, that use different interpretations and assumptions of the same underlying physics but which predict the same output variables. Additionally, different models can be combined in predictive cascades (e.g. coupling climate and hydrological models to project future flooding). These cascading, integrated models (which must also allow for uncertainty) are used in decision and policy frameworks. Ensemble techniques are important to manage model uncertainty (driven by the chaotic nature of natural systems and lack of data for model initial conditions), a key issue in Environmental Science.

Moving environmental models to cloud architectures has the potential to revolutionise Environmental Science through supporting an open, collaborative and integrative approach. This area, however, is in its infancy with only initial insights gained through projects

such as EVOp [1]. Although there have been initiatives to develop 'community' models in some domains, there has been no research into raising the level of abstraction by using cloud services. In order to achieve a status of 'models in the cloud', it is necessary to significantly raise the abstraction of the underlying cloud services, to manage the distributed computation and allow scientists to freely operate in their specific domain. Therefore, a real opportunity for SE to provide high-level support for cloud execution of a large and diverse range of environmental models.

2.2 Environmental Models and Cloud Computing

Cloud computing is a technological paradigm evolved over time from existing technologies like grid computing, utility computing, virtualisation and autonomic computing [6]. Cloud computing supports the provisioning of on-demand, varying levels of services over the internet such as Infrastructure as as Service, where users can choose from a heterogeneous pool of physical resources, Platform as a Service offering platform level software and Software as a Service. Cloud computing has opened up a world of new opportunities with its qualities such as scalability, elasticity, reduction in capital expenditure, on-demand unlimited resources, heterogeneous options, virtualisations and pay per use faclities. Large companies such as Amazon (AWS), Google, Microsoft and others are striving to provide reliable and cost-efficient cloud platforms to users.

Exploration of cloud computing services in ES is an under-research area. The climateprediction.net (CPDN) project started as an initiative to address the uncertainties associated with climate model parameters [10]. This project is currently running on the BOINC framework comprised of volunteer computing resources. Considering the limitations of the BOINC based infrastructure in terms of scaleability, anticipation of performance and control over resources lead the researchers to start looking for cloud offerings, specifically AWS at considerable effort.

In research related to Numerical Weather Prediction, researchers at NASA's Marshall Space Flight Center and Ames Research Center collaborated on the use of the WRF model by making use of public and private cloud resources [9]. STRC-EMC was a case study deployed on AWS EC2 and discussed the opportunities enabled through the usage of cloud resources, especially for those who have no access to large scale HPC infrastructure. The research emphasised the on-demand nature of cloud computing and the use case of running simulations if HPC access is lost in a disaster.

Similarly, Chen et al. [2] evaluated the performance and reliability of running the Community Earth System Model (CESM) on AWS EC2. CESM is a widely used climate model developed by the National Center for Atmospheric Research (NCAR). In this experiment CESM is deployed on the StarCluster which is an open source cluster computing toolkit for creating a cluster on AWS EC2 nodes. Comparable results were seen when compared to the simulation execution time on the local HPC. In contrast they found the black-box nature of cloud services can result in variable performance. However, scaleability and creation of reusable machine images was considered a useful feature for climate scientists.

[1] http://www.evo-uk.org/

2.3 Software Engineering and Environmental Science

In his call to action for software engineers, Easterbrook [3] specifically identifies "Computer-Supported Collaborative Science" as a fundamental way in which software engineering can contribute to addressing the grand challenge of climate change. He identified that through supporting earth system models with software engineering tools and techniques we can accelerate the process of getting scientific ideas into working code. In his keynote at MODELS'16 [2], Blair specifically identified the Model Driven Engineering (MDE) community of software engineers as having the tools that could revolutionise the way Environmental Modellers work, through abstraction of complex computing systems, allowing them to concentrate on the science. Through this work we are responding to these calls.

Easterbrook and Johns [4] performed an eight-week observational study at a national weather forecasting agency, and comprehensively documented the software engineering practices of the organisation and structure of the models in development and use. In our study we take a broader perspective and interview individual scientists from academic and government research institutes about a broad range of environmental models. Easterbrook and Johns found environmental model coding is usually done in-house by domain experts. These experts typically do not have SE backgrounds, and perform model coding in addition to their day jobs, i.e. performing science [4]. The typical structure for large models consists of a small number of code owners and a larger set of contributors [4], often called "community models". For example, global climate models (e.g. CESM [3]) are typically split into components that represent both physical boundaries and the different research communities, i.e. ocean and atmospheric science. The development of these large, multi-faceted community models can take decades and involve hundreds of scientists. Consisting of millions of lines of code in some cases, disentangling their structure in a SE perspective can be challenging [1]. Here, Easterbrook argues for the engagement of the software community as they ".. bring a unique set of skills related to the analysis and (re-)design of complex technical systems" arguing that "software and computational thinking are critical components of the solution" [3].

There has been work towards implementing DSLs and MDE in other scientific disciples, for example Whittle et al. [14] generated Fortran code from a high level graphical specification, composing code from a subroutine library. However in environmental modelling, there are no such examples of languages orchestrating systems from components in this way.

3 CYCLE 1: INTERVIEWS WITH ENVIRONMENTAL SCIENTISTS

In our first project cycle we held a series of interviews with environmental scientists with the aim of gaining an understanding of practices, including software engineering practices in developing and deploying environmental models. This is the first cycle of user engagement and embedding domain experts in the project.

[2] https://www.youtube.com/watch?v=2T11Gpq0PEg
[3] http://www.cesm.ucar.edu/

3.1 Selection of Participants

We selected the interviewees from a broad range of environmental disciplines to gain experience of a wide range of practices. They were chosen specifically as a group that develop or use computer based models of the environment. Five interviewees gave us a range of perspectives and was achievable in the time available, although we note the low number means the findings are not statistically significant and may be subjective. There was a mix of academic (A,B) and independent research institutes (C) involved. The participants come from partner institutions and departments on the same campus, and had never worked with the interviewers before. Some details are listed below, names have been changed:

- TERRY (A)- Completing PhD student, using a community developed global climate model (CESM) to conduct experiments.
- GEORGE (C)- Statistician, developing their own statistical model of insect populations.
- CLAIRE (C)- Quantitative Ecologist, tasked with connecting models that were never designed to be connected.
- ROLAND (A)- Postdoctoral Researcher, using a community weather model (WRF) to study urban heat islands in Africa.
- TATIANA (B)- Collaboratively developing a soil chemistry model in a multi-site team.

3.2 Interview Technique

The interviews took the form of a semi-structured conversation about the interviewees work - we asked them to introduce and demonstrate the model they work with / on, show us code and scripts they had developed themselves and run the model. Additionally, we prompted the interviews using the question framework shown below if it was not covered naturally in conversation. The interviews lasted between 1.5 and 3 hours.

- **Individual** - the types of model the individual works with and how frequently they work with models. If they write new models or adapt existing models, which programming languages / tools / frameworks are used. The individual's SE background and experience.
- **Model specifics** - how the model runs and is deployed. Include information related to the architecture and computational requirements of the environmental model. The architecture specifies if the model is designed as a stand-alone module or composed of other integrated or coupled methods/models.
- **Automation** - captures information related to the availability and description of work-flows, scripts or GUI to run that model.
- **Architecture** - describes the computational architecture the model is running on, i.e distributed, shared, cluster based, etc.
- **Data Formats & storage** - informs about the type and formats of input and output data required to execute a model simulation. It also details any hardware and software related requirements for storing the input and output data.
- **Pre-processing & deployment** - keeps track of any requirements or programming tasks necessary as pre-requisites related to data, model or model deployment.
- **Processing** - informs about the duration of different model simulations and the general practices the scientist follows to keep track of running simulations.
- **Analysis & visualisation** - describe the software tools used by the scientists in order to analyze and visualize model outputs from the data produced.
- **Limitations & enhancements** - discuss general practices that scientists follow and what challenges they are faced with, and to identify different aspects of human values.

Two researchers visited the interviewee in their normal work place and recorded the conversation and screen interactions using a video camera. The interviews were transcribed and we took a grounded theory approach to coding, here individually coded by the two researchers. At the first pass the researchers met to agree on the coding framework to use, and findings are presented below grouped by these themes.

3.3 Findings

The interviews provided the researchers with rich insight into the software engineering practices engaged by our group of environmental scientists, and the variety of approaches taken in running and engineering models.

3.3.1 Technical.

Computational Demands and Resources. The resources required to run models generally varies with the number of runs, resolution of the grids and geographical areas. Although they can be compiled for local machines, global climate and weather models are run on HPCs by our interviewees. TATIANA, GEORGE, and CLAIRE usually develop their model runs on desktop computers, and each encountered problems with transferring the models to HPC. Scaling up to HPC is required if additional processing power is required, for example when performing many parallel runs of a model, or running the model at a much higher spatial resolution. TATIANA's Matlab model required some additional skillsets to deploy on the HPC. One of the main challenges for CLAIRE's experiment was getting a pre-compiled model to run on their multi - PC setup: "The main challenge we had, which seems ridiculous was that we couldn't get the DLL to work on a machine that wasn't the one it was compiled on for 3 or 4 months, because the person that set it up didn't know how to compile it to run on other machines". None of our interviewees used models or services deployed in the cloud directly in their environmental modelling work.

For our interviewees, data exchange was performed with flat files and FTP was used to move the data between individuals local computers and HPCs.

For CLAIRE, there was a miscalculation or perhaps a lack of recognition of the value of computing expertise in the project: "[They thought] both the time for the ICT people downstairs and the computing resources were free ... they weren't".

Computational Skills and Expertise. Each of the interviewees had self-taught themselves the system administration and programming skills required to install, develop and deploy their models. None had a formal education in computer science or software engineering. CLAIRE: "I've never been taught to do anything on a computer apart from at school", TATIANA: "... my training in programming is pretty limited. It's more just I've picked it up as

I've gone on". For example there was little understanding of object oriented programming, or attempts to design systems or software architectures before building models amongst interviewees.

There was a shyness in showing the interviewers code they had written TERRY: "Go easy on my scripts, I'm just learning", and hints at an understanding of a missed opportunity GEORGE: "... there's just a tendency to jump in and say, "I've got the data, let's start." By the time you've written ten lines of code, that quickly becomes a hundred, it quickly becomes, "Well, we can't start again now.""

For ROLAND, the major challenge in running experiments was installing the model - as this required systems administration skills such as installing dependencies and navigating command line tools.

CLAIRE had called on a member of an informatics team in her organisation to help deploy the model developed to multiple physical nodes, but was not able to use their skills at much as desired as their time was not budgeted for in the original project proposal.

Code Understanding. The programming and scripting languages the models our interviewees are written include scientific and mathematical languages such as Fortran, Matlab, and R. In the case of TERRY and ROLAND Bash scripts are written to prepare data and deploy the models they work with. Some general purpose languages including Python are sometimes used in the analysis of output data. The choice for which language and environment used falls to familiarity TERRY: "I've got many, many scripts written and created in Bash. So, if I want to do something else I just look those previous scripts and just modify a bit what I need to do" but there is an awareness that other options are there GEORGE: "I'm doing some things in R here that would probably be more efficient to do in Python".

The programming language can be a barrier for the scientist to understand the science of the model CLAIRE: "I really don't know what's going on in it because I did not write it... it took me ages to read through the Fortran file because I don't read Fortran really".

Whereas GEORGE prefers the configurability of Notepad++ to an IDE, IDEs are used to develop and also run "local scale" models- R Studio used by CLAIRE, Matlab, visual studio and xcode by TATIANA, but world weather and climate models require the use of command line skills and bash scripts to deploy (ROLAND, TERRY), there is no integrated environment or framework used for those.

Version Control. GEORGE described using Git for his model code, but offered some insight into why others don't use version control when working alone on models: "I think if there's only you that works on it, there's less of an impetus for putting it in Git". Similarly CLAIRE has recently started to use Git in preference to change log headers in source code files, but describes why people are hesitant: "I mean definitely a bit of a cultural thing about we don't really have a good version control culture ... some of them have never heard of [version control]". TATIANA describes one problem of working in multi-site teams without version control: "I don't fully understand why some bits are changed, or how they're doing what he says they're doing", and GEORGE the recent situation before implementing version control: "everyone would have slightly different versions because you have this thing of Chinese whispers". TERRY manually documents changes in headers at the top of script files - "This is the way I document when I do this. When I write the script, and then, if I add any change I keep tracking that. Then, a description, what the script does". Similarly ROLAND does not use version control system for his scripts.

We asked if the interviewees implemented any method of versioning model run outputs- GEORGE: "No, I don't have any versioning of outputs, and in fact I have got into a bit of a mess with that, actually ... Obviously, when you run this you're supposed to name these yourself". TERRY archives what he considers to be useful to a portable harddrive, "... when you are doing a study, later you usually regret not saving more data", but when publishing articles "They keep the data publicly available with DOI and with that people can retract the data and check whether my study is correct or do something else".

Fault tolerance / resilience. The community climate model and weather models are relatively fault tolerant, allowing the scientist to restart where the model stopped. However crashes or faults in these models are usually due to invalid boundary conditions or similar, indicating a problem with the way the model was set up. CLAIRE struggled with one model as it would crash apparently randomly without warning. This would output data as it went allowing some data to be used. The model would have to be manually restarted from the time of failure to continue, it would not recover automatically.

(In)efficiency. There is some inefficiency in some tasks, for example in the large volume of data exchanged between computer systems, and having to download large flat files to extract a smaller amount of data which the environmental scientist is interested in. In terms of architecture, GEORGE highlights that a single language, single environment approach may not be desirable- GEORGE: "This whole thing is written in R, and that is optimal for the statistical modelling bit because that's what R's good at, but for a lot of the other things, it's not ideal. For a lot of that spatial intersections and whatnot, I'm sure Python is more efficient".

When programming, code reuse through building functions and modules is sometimes constrained by the environment or familiarity with the environment -GEORGE: "But when I write in R, I'm more like 'follow the trend' and write a straight line of code. And if you need to repeat it then copy and paste. After some time, I really got frustrated and I was more like, no. If I need to reuse, there should be some way that I call simply, just as I do in the Java". GEORGE goes on to hint at the improvements that could be made through working with software engineering: "if you knew that from the outset and you planned it, you might say, 'Well, what we need is, we need a person who could do this and could easily do something, write something, write some Python code or whatever, do that.' And then all we'll have to do is link that to this and sort of sketch it out. And we never did that ... we'd just dive in".

Compatibility. Interfacing of data amongst our interviewees is always done with flat files through CSV files, netCDF files and Excel documents. The models they demonstrated to us do not interface to other types of data store or have other interfaces beyond command line or IDE deployment and yet there is a growing need for this. Coupling models is rarely done, and is a real challenge. CLAIRE required a regression function to interface models "There was no design at the start for them to be brought together ... we put in place some simple regression that sits in between the two models and acts as a kind of calibration".

TATIANA would like to use other models in her work, but interfacing them is a concern "... there's lots of candidates out there that you could use rather than creating a new model. Then it's things like what programming language are they in? Are they going to mesh with [model name] very easily?".

3.3.2 *Scientific.*

Model Understanding. The interviewees all have a high level of understanding of how the models they are using work. However the depth of understanding varied across the set of interviewees and the teams which they work with. CLAIRE: "... brought together they were in different languages and then the process space modellers do not really understand the statistical model and vice-versa", and TATIANA "I'm a geographer not a mathematician but I could follow them".

Uncertainty. Modellers deal with uncertainty in their models in different ways. Each global climate model run produces a different output, and it is up to the Environmental Scientist to design ensemble model runs to explore uncertainty arising because of this variability TERRY: "Greenhouse gas concentrations, aerosols, the temperature of the ocean is the same so it's going to give you something different because the chaos of the system is going to give you some kind of variability". Sometimes models are designed to capture and preserve this uncertainty GEORGE: "What you actually get at the end is a map, so there's two maps here. We've got the actual thing on the left, the bees, and then the uncertainty associated with that. We were quite keen to keep the two together so this idea that we always want to present the uncertainty. This is a statistical model, so there's always uncertainty associated with it. The output is preserved and the two are always linked. In the NetCDF file, as well, both the estimate and the uncertainty are in the NetCDF file". But this methodology does not extend to source data GEORGE: "[..] in statistics, that's often ignored, that you assume the data you're dealing with is the truth. That's not always the case, obviously. Yeah, the uncertainty here is the uncertainty as a result of that modelled relationship, not [the source data]".

Dealing with uncertainty is something environmental scientists would like improve. One barrier is the complexity of the computation required to track it GEORGE: "... keeping track of uncertainty, ... making sure it's propagated through, is sort of easy technically because you could just say, well, what we're going to do is we're just going to take samples, we're just going to do a sort of Monte Carlo re-sampling thing and just put it through each component of the model a hundred times and let that work its way out in the end. Which is sort of easy enough to do, but that's computationally where things get very complicated". CLAIRE: "And it takes 5min for each square (there are 200,000) so even doing the uncertainty analysis on that it would take us years, so we cannot do that".

Data Sorting and Extraction. Data preperation is a significant, sometimes complex and key task performed by our interviewees. The volume of data needed for input and generated by the models is often significant (terabytes) and usually downloaded locally, processed and re-uploaded. In the case of the climate model, the data is in the form of flat files that need to be downloaded, processed and re-uploaded -TERRY: "... first we look through raw files, then extract the variables we need via NCO commands, and that is very simple ... using this file and put it in this new file.... Instead of having terabytes, you'll have gigabytes, and that will help ..." this work is done by a script the scientist wrote themselves with their own strategy because the tools aren't available, or shared to do this GEORGE: "I didn't find any standard functions to do this. I adopted my own version".

Sometimes simple tasks like unit conversion are done manually in data preparation scripts - GEORGE: "... rainfall is in an odd unit like kilogrammes per metre squared per hour. ... I just wanted millimetres, and so yeah, there was a bit of a conversion ... That was just a case of multiplying it out, but there's not many R-libraries for specific manipulation like that ... ". Translating resolutions of grids and data is also done independently CLAIRE: "Oh this is just a bit silly, we wrote it very inefficiently just translates grid references in a very long way into that." ROLAND: "... it was LiDAR, kind of satellite data product to get all the building heights for the UK ... Because the model I'm running is a kilometre resolution, so I don't really want individual building heights. I was kind of aggregating those two, at a kilometres resolution. So, I did that for the model".

None of our interviewees described their models interfacing with data sources other than flat files, such as netCDF and CSV. TATIANA: "I'll show you this in the Matlab, basically there's a CSV file for each plant functional type, which has ... Plant parameters for each plant type". One institution had a data store that serves flat files, and these did not have well documented meta data GEORGE: "The data is numeric on 1-23, but they're actually classes, it's a classification, so 1 actually means broadleaf woodland. You've got be careful that you acknowledge that that isn't a numeric, which from my understanding it's not obvious in a netCDF ... That information unfortunately is not in the netCDF".

3.3.3 *Human.*

Time Management. Most of the interviewees work on one project at a time, but some are split across multiple projects using different models - CLAIRE "i might only work on 3 or 4 on an average week".

Ownership - GEORGE described issues with agreeing sharing data across project partners "While they have the data, they're not going to let it out the door" and "the computation aspect wasn't so much of an issue, as much as just a space where everything could be, the data, the code, and whatnot, and there was some sort of control over what versions what people were using". The ownership of a model is sometime maintained and access controlled by individuals or research groups and in some cases it is undefined - TATIANA: "So, this model was [written by] a guy called Terry... and then Fiona was working on it. I say Terry; there was other people as well, but I think he was the leader of the paper. And then Fiona started working on it in combination with these people at [another institution], and they added ... to it.... So, it's still evolving". There is a recognition that the models produced and used can go on to influence big decisions - CLAIRE "The idea is its all very policy relevant. It is always little bit scary. But then what's we do".

Multi authorship. TATIANA described problems with sharing the development of a model "...I'm having issues because he's changed variable names, and things like that." and using different naming conventions "Rather than cooling and plant functional types he calls them land use parameters".

3.4 Focal Point: Insight and Opportunity

After the interviews cycle was conducted, the research team regrouped at this focal point and summarized the key opportunities and insight they found from observing the way the interviewees worked and the tools that were used.

3.4.1 Insight from Interviews. The findings above clearly draw parallel to the opportunities for SE to engage with scientists highlighted by Easterbrook [3], however without influence from this the research team looked for cross cutting themes in the interview data that were present at all levels of model development, summarised here:

- A high level understanding of systems administration skills is required when installing, configuring and deploying models. This understanding is self taught.
- Data is stored as it "always has been", in flat files and exchanged via FTP. Data is prepared locally then uploaded to the processing computer (usually HPC facility).
- Alternative architectures are rarely considered or understood - we observed models written on a local machine, then adapted at some effort to run on HPC. HPC may be the best architecture for some core computations, but many other tasks may be better suited to other architectures
- Models are often monolithic written in a scientific language, restricting the ability for each component to be deployed to appropriate compute architecture.
- Models are generally not developed with interfacing or coupling to other models in mind, however this is becoming an increasingly common task.
- Implementation of version control supports collaboration and understanding of changes.

3.4.2 Opportunities for SE in ES. These themes can be translated into the following opportunities for SE in environmental science.

- **Abstraction.** There is a real opportunity to raise the level of abstraction and allow scientists to deploy models without needing to become systems administrators. Model Driven Engineering provides the tools through Domain Specific Languages to be able to do this.
- **Framework Support.** Build frameworks and environments that support separation of the scientific tasks from data preparation, allowing each component to be developed in appropriate languages and deployed to most appropriate architectures. Defined interfaces will allow models to connect to other models, "queryable" data sources (rather than flat files) and other services. Support scaling of models from desktop to cloud and HPC.
- **Flexible Architectures.** These frameworks and DSLs can facilitate deployment to new and emergent architectures, allowing scientists to take advantages of on demand compute in the cloud and alternative architectures such as GPU nodes.
- **Education.** None of our sample of interviewees had any formal software engineering training. More understanding of software engineering by the model developers would help them to structure developments in more scale-able, reusable, fault tolerant manner.

4 CYCLE 2: AUTOMATED DEPLOYMENT OF WRF TO CLOUD ARCHITECTURE

In the second cycle we address the first two of the opportunities highlighted in the previous section through cloud deployment, experimentation and feedback from an expert group. We create an easy to use cloud deployment of the WRF weather forecasting model, and this provides insight into DSL devlopment. Here, we raise the level of abstraction through reducing the complexity of deployment and leveraging the on demand scale-ability of cloud architecture. At each subsequent cycle we will take our latest work to a relevant user group to feedback and inform development.

This section describes the WRF model, cloud deployment, scripts developed and the feedback from an expert users group at a "show and tell" focal point marking the end of project cycle 2.

4.1 The WRF model

The WRF model [13] is a large community-based endeavour (around 40,000 users), supported by NCAR. The model is primarily used for atmospheric research and forecasting across a wide range of scales (thousands of kilometres to meters). The diverse range of extensively validated science WRF can simulate includes regional climate, air quality, urban heat islands, hurricanes, forest fires, and flooding through coupling with hydrological models.

WRF takes the initial state of the atmosphere and propagates this forward spatially and temporally by numerically solving (and conserving) equations of mass, momentum and energy. A suite of parameterisations is used to account for sub-grid scale or processes too complex to be accounted for by the model, e.g. radiation. The model is typically run for short time periods, days to months and uses a nested domain structure is used to increase horizontal resolution over the required study area, i.e. to limit unnecessary increases in computational time.

WRF is chosen as a case study here for the following reasons: i) WRF installation is viewed as a barrier to use; ii) cloud resources will enable WRF users to conduct simulations beyond current capability [12]; iii) WRFs open-source nature and portability; and iv) the benefits SE will provide to WRFs large community user base. The goal is to remove time-consuming user computing difficulties that that could otherwise be spent on core environmental science.

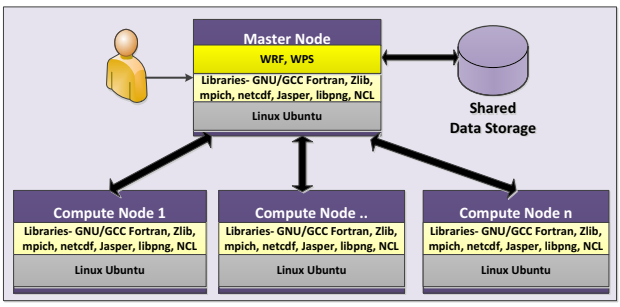

Figure 2: MPI cluster on Microsoft Azure IaaS.

4.2 WRF Cloud Infrastructure

We deployed WRF on a Microsoft Azure cluster, the basic architecture of this cluster is shown in Figure 2. The architecture of the WRF model itself is complex, described by [13]. We deployed WRF to a Message Passing Interface (MPI) supported cluster composed of 9 standard compute nodes from the Microsoft Azure Dsv3-series each having a 3.2 GHz Intel Xeon E5-2673 v4 (Broadwell) processor. Of the 9 nodes, 1 node is a master node taking care of all the compilation and providing a means of sharing the storage and computation with all the nodes. The computational specification of cluster nodes are described by Table 1.

We used a predefined image of Ubuntu Server 16.06 LTS for each of the cluster machines and each node has 8 processors with 32 GiB RAM and temporary storage of 64 GiB that is considered a secondary storage for each compute node. We used the GNU Fortran and GCC to compile WRF. The cluster provides primary storage of 100 GiB shared amongst nodes via the Network File System (NFS). The shared location contains all the simulation related input/output data and files required for WRF configuration as well as compilation. All the cluster nodes and storage are deployed in Western Europe under one secure virtual network and have friction less access to enable data sharing and execution of MPI based jobs.

4.3 Cost & Performance Insight

In order to gain insight into the cost and performance of WRF deployment, the same simulation was run on a HPC cluster[4] and the Azure cluster using different numbers of nodes. It is not expected to achieve comparable performance since the Azure cluster is composed of ordinary general purpose nodes rather than dedicated HPC supported nodes with high infiniband access. Rather, the intention is to use this performance data to explore use cases in which cloud deployment on general purpose nodes is appropriate. No GPU nodes are used here.

A 24-hour WRF simulation was centred over the Lagos, Nigeria metropolitan area using WRF version 3.9.1.1. The reason for choosing this location was interest in the rapid urbanisation the city has undergone in the last few decades and its impact on local climate. The simulation used four nested domains, with increasing horizontal resolution (3:1 ratios) from 27 km in the outer domain to 1 km in the inner domain. The domains each contained 91 x 91 grid cells, i.e. covering a horizontal distance of 2500 km in the outer domain and 91 km in the inner domain. The model was configured with default physics and parameterisations, with exception of setting the single-layer urban canopy model [8] for the urban physics. The free-to-use NCEP (National Centers for Environmental Prediction) 6-hourly, 1-degree horizontal Final Operational Model Global Tropospheric Analyses data were used for initial and lateral boundary conditions.

The results of running simulation on HPC cluster as well as Azure with varying numbers of processors are shown in Figure 3. A clear reduction in execution time is seen for both HPC and Azure as the number of processors is increased from 8 to 64. This indicates that a cluster composed of ordinary general purpose compute nodes can also be used to effectively run WRF simulations.

[4]http://www.lancaster.ac.uk/iss/services/hec/

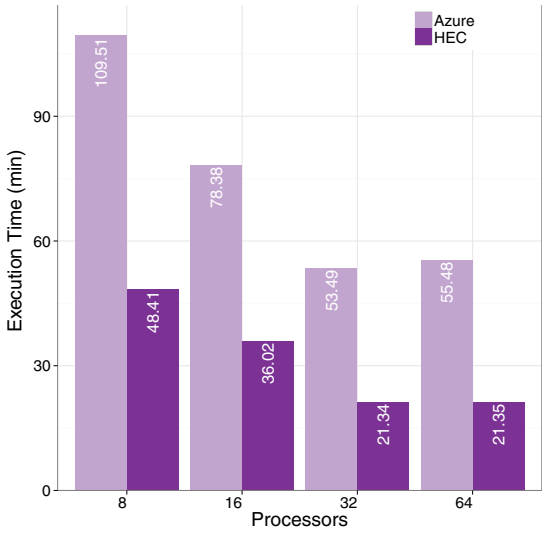

Figure 3: WRF simulation execution time over Azure cloud and HEC.

Another experiential insight is seen when the price is calculated for running the same simulation on different numbers of processors on the Azure cluster. Table 2 shows the cost of running these simulations, where minimum compute cost is less than a dollar and maximum is approximately equal to $3. This shows a potential benefit for the scientist who do not have access to high performance computing infrastructure (or does not want to wait in the queue) and can run their models within affordable budget. The cost of a dedicated HPC cluster is not considered in this work, a per-simulation price would be almost impossible to calculate since it is dependant on so many factors. Instead we focus here on the opportunities cloud technologies bring above a HPC offering.

These results show the flexibility of cloud deployment in terms of speed and cost. Surprisingly, the execution time of running the model on 32 and 64 processors is almost the same, however 64 processors have a higher cost. This indicates that in WRF there is an optimal threshold of assigning number of processors to a simulation depending on simulation size. Finding this optimal deployment is subject to future work.

4.4 Qualities of Cloud Deployment

Here we discuss some of the qualities of the cloud deployment.

Raising the level of abstraction. The deployment complexity is reduced by abstracting away complex procedures of WRF installation and deployment. This is done by providing executable scripts that can take care of all the dependencies starting from downloading libraries to setting up environment for running a WRF simulation. The scripts are able to deploy WRF on a private Linux machine or a virtual machine on some public cloud provider's infrastructure. A walk though guide is also provided for enabling end users to create a compute cluster on Microsoft Azure's infrastructure for WRF deployment without being an expert in the cloud computing

Table 1: Computational specifications of Azure Compute instances.

Series	Instance Type	Cluster Node	vCPU (GiB)	RAM (GiB)	Storage	IOPS/MBps	Expected N/W Performance	Price ($/h)
General Purpose	Standard D8sv3	Master x 1	8	32	64	12,800/192	4/High	0.357
		Compute x 8	8	32	64	12,800/192	4/High	0.357

Table 2: Cost and performance of running WRF simulation on Azure cluster.

X Processor Cluster	Execution Time (minute)	Cost/hr (/node)	Total Cost ($)
8 (1x8)	109.51	0.369	0.738
16 (2x8)	78.38	0.369	1.47
32 (4x8)	53.49	0.369	1.44
64 (8x8)	55.48	0.369	2.952

domain. The user is also provided with the necessary steps to follow for creating machine images that can increase the deployment progress and enhance reproducability.

Leveraging cloud services. A user can have access to on-demand resources with no upfront cost, where they can choose from a pool of heterogeneous options providing different virtual machine configurations. Here, a user can save HPC queue time by running a simulation on cloud infrastructure. Additionally, clouds can offer data analytics, machine learning and container fabrication services.

Democratising models. Providing the mechanisms to deploy environmental models in cloud environments can open up access for all. Not just well funded research institutes with their own HPC resource, anyone interested would be able to deploy models and run experiments in the cloud.

Performance and cost. The performance of the HPC always outstripped the cloud deployment on standard nodes. As cloud services are developed this performance gap may change. Cloud deployment is "on demand" with no queue time required, so experiments can be run instantly. A "per use" cost enables flexibility, but may not be a comparable cost when comparing overall performance and usage of a HPC.

4.5 Focal Point: Feedback from expert WRF users

In the cycle 2 focal point the cloud deployment was demonstrated to a group of 6 expert regular WRF users in a "show and tell" event. Here we held a brainstorm session to identify appropriate use cases for cloud deployment of WRF and identify potential end users to get engaged in developing the DSL for deployment.

All users present regularly run research experiments using WRF, three were involved in some kind of model development, three were involved in teaching and one was most interested in using the output from WRF as input to other models.

We first described the project, and our outline plans for the future. WRF was deployed during this time using the unattended automated script. We then held a brainstorm and discussion session to get feedback on this mechanism of deployment and the experts proposed likely situations where cloud deployment would be useful (use cases), the impacts / benefits to their work and where it may not be appropriate to deploy in the cloud.

Immediately the experts saw the removal of a key barrier to use of WRF by the simplification of the installation process. Through discussion the following use cases for WRF in the cloud emerged:

- A) New users. For example, Masters level students who usually take 3 months to learn how to install and configure the model before doing any science.
- B) Those who just want to run the model in a standard way and get some results to feed into other models or projects.
- C) Power users who may want to quickly deploy for a project, perhaps in a parallel execution in the cloud without wanting to wait for a HPC queue.

A future DSL may need to cater for these users, however the "power users" will likely be more familiar with navigating command lines and scripting so may just need guidance on cloud deployment, or a DSL for architecting this deployment.

The WRF experts were excited by the potential of connecting WRF to other cloud services, such as data stores, machine learning and analytics etc. and leaving the data in the cloud to reduce the amount of data transfer, taking the data to the services.

The people in the room were also very interested the concept of developing models from the outset as "micro services", with the intention of reusing services in other ways.

5 CYCLE 3: FUTURE WORK- DSL DEVELOPMENT

From our expert user group, we distilled 3 interesting avenues of enquiry for our work:

- 1) Develop a DSL for cloud deployment of WRF with the 3 user groups identified.
- 2) Connect WRF cloud to wider cloud services - for example connect the output to a useful data store that can be queried e.g. geosparql and analytics services.
- 3) Work with a model built from "scratch", and build it using micro services orchestrated by a development of the DSL.

Cycle 3 will address the first point from the above. We will do this by forming user groups from each of the use cases and collaborate by embedding users in fortnightly cycles of DSL design and development. A focal point will be formed by meeting with the expert users in another Show and Tell event.

To further abstract from underlying computing, WRF and the automated script will be wrapped in a docker container, with a DSL designed to configure this deployable container and the architecture to which it is deployed. This will be configured by the

requirements of the scientist's experiment and the constraints (time, cost), themselves configured by DSL.

6 CONCLUDING REMARKS

This work seeks to raise the level of abstraction for environmental scientists to allow them to concentrate on their science rather than configuring supporting computing systems. We do this by engaging with scientists to understand their domain and produce technologies that are fit for purpose. In doing so, we wish to leverage new architectures to facilitate deployment, and to interface environmental models to each other and to the wider service fabric in the cloud such as data stores and analytics engines.

In this paper we presented findings from a series of interviews with environmental scientists working with different kinds of models, unveiling the way in which they work and the degree of familiarity they have with systems administration, software engineering and understanding of compute architectures. We found a high degree of system administration is required, data is not stored efficiently and alternate architectures are not used. Programming is monolithic, without defined interfaces, and there is little a thought to code or service reuse.

We find opportunities for SE to engage with ES, to educate in methods and techniques, and highlight the opportunities for abstraction and framework support to provide tools to enable scientists to leverage flexible architectures. This would allow more models to be connected and run more efficiently, and to allow scientists to concentrate on the science, resulting in a better understanding of phenomena and uncertainties which would hopefully be reflected in better policy informed by results.

Toward this, we built an automated cloud deployment of a complex weather model, which experts could see instantly would break down barriers to entry of new scientists, and allow connection to cloud service fabrics. This paves the way for future work developing DSLs to allow scientists to describe experiments in their own language and abstract from the systems configuration and complex deployment required by models.

Supporting this work is the Speedplay methodology [5], rooted in participatory design that sees the computer scientists and environmental scientists in equal partnership, developing together. In this research we will go full circle, not just unravelling and understanding the opportunity, but designing and building technologies with end users embedded in the process. This will give the resultant tools and techniques the best chance of being adopted and will inform the future development of environmental models.

We have seen that despite doing a lot of software engineering and computing, Environmental Science as a discipline has not kept pace with advancing technologies in these fields. There are opportunities to work with Environmental Scientists to make a real transformative difference in the discipline, and help them to better understand our environment and help policy makers to make the right decisions for the future of our planet.

ACKNOWLEDGEMENTS

We thank the interviewees and the team at Ensemble[5]. This work is funded by EPSRC Grant EP/N027736/1: Models in the Cloud: Generative Software Frameworks to Support the Execution of Environmental Models in the Cloud and supported by a Microsoft Azure research grant.

[5]https://www.ensembleprojects.org/

REFERENCES

[1] K. Alexander and S. M. Easterbrook. 2015. The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations. *GEOSCIENTIFIC MODEL DEVELOPMENT* 8, 4 (2015), 1221–1232. https://doi.org/{10.5194/gmd-8-1221-2015}

[2] Xiuhong Chen, Xianglei Huang, Chaoyi Jiao, Mark G. Flanner, Todd Raeker, and Brock Palen. 2017. Running climate model on a commercial cloud computing environment: A case study using Community Earth System Model (CESM) on Amazon {AWS}. *Computers Geosciences* 98 (2017), 21 – 25. https://doi.org/10.1016/j.cageo.2016.09.014

[3] Steve M. Easterbrook. 2010. Climate Change: A Grand Software Challenge. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research (FoSER '10)*. ACM, New York, NY, USA, 99–104. https://doi.org/10.1145/1882362.1882383

[4] Steve M. Easterbrook and Timothy C. Johns. 2009. Engineering the Software for Understanding Climate Change. *COMPUTING IN SCIENCE & ENGINEERING* 11, 6 (NOV-DEC 2009), 64–74.

[5] Maria Angela Ferrario, Will Simm, Peter Newman, Stephen Forshaw, and Jon Whittle. 2014. Software Engineering for 'Social Good': Integrating Action Research, Participatory Design, and Agile Development. In *Companion Proceedings of the 36th International Conference on Software Engineering (ICSE Companion 2014)*. ACM, New York, NY, USA, 520–523. https://doi.org/10.1145/2591062.2591121

[6] I. Foster, Y. Zhao, I. Raicu, and S. Lu. 2008. Cloud Computing and Grid Computing 360-Degree Compared. In *2008 Grid Computing Environments Workshop*. 1–10. https://doi.org/10.1109/GCE.2008.4738445

[7] J. E. Kay, C. Deser, A. Phillips, A. Mai, C. Hannay, G. Strand, J. M. Arblaster, S. C. Bates, G. Danabasoglu, J. Edwards, M. Holland, P. Kushner, J. F. Lamarque, D. Lawrence, K. Lindsay, A. Middleton, E. Munoz, R. Neale, K. Oleson, L. Polvani, and M. Vertenstein. 2015. THE COMMUNITY EARTH SYSTEM MODEL (CESM) LARGE ENSEMBLE PROJECT A Community Resource for Studying Climate Change in the Presence of Internal Climate Variability. *BULLETIN OF THE AMERICAN METEOROLOGICAL SOCIETY* 96, 8 (AUG 2015), 1333–1349. https://doi.org/{10.1175/BAMS-D-13-00255.1}

[8] Hiroyuki Kusaka, Hiroaki Kondo, Yokihiro Kikegawa, and Fujio Kimura. 2001. A Simple Single-Layer Urban Canopy Model For Atmospheric Models: Comparison With Multi-Layer And Slab Models. *Boundary-Layer Meteorology* 101, 3 (01 Dec 2001), 329–358. https://doi.org/10.1023/A:1019207923078

[9] Andrew L. Molthan, Jonathan L. Case, Jason Venner, Richard Schroeder, Milton R. Checchi, Bradley T. Zavodsky, Ashutosh Limaye, and Raymond G. OâĂŹBrien. 2015. Clouds in the Cloud: Weather Forecasts and Applications within Cloud Computing Environments. *Bulletin of the American Meteorological Society* 96, 8 (2015), 1369–1379. https://doi.org/10.1175/BAMS-D-14-00013.1 arXiv:https://doi.org/10.1175/BAMS-D-14-00013.1

[10] D. Montes, J. A. Añel, T. F. Pena, P. Uhe, and D. C. H. Wallom. 2017. Enabling BOINC in infrastructure as a service cloud system. *Geoscientific Model Development* 10, 2 (2017), 811–826. https://doi.org/10.5194/gmd-10-811-2017

[11] J. M. Murphy, B. B. B. Booth, M. Collins, G. R. Harris, D. M. H. Sexton, and M. J. Webb. 2007. A methodology for probabilistic predictions of regional climate change from perturbed physics ensembles. *PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A-MATHEMATICAL PHYSICAL AND ENGINEERING SCIENCES* 365, 1857 (AUG 15 2007), 1993–2028. https://doi.org/{10.1098/rsta.2007.2077}

[12] Jordan G. Powers, Joseph B. Klemp, William C. Skamarock, Christopher A. Davis, Jimy Dudhia, David O. Gill, Janice L. Coen, David J. Gochis, Ravan Ahmadov, Steven E. Peckham, Georg A. Grell, John Michalakes, Samuel Trahan, Stanley G. Benjamin, Curtis R. Alexander, Geoffrey J. Dimego, Wei Wang, Craig S. Schwartz, Glen S. Romine, Zhiquan Liu, Chris Snyder, Fei Chen, Michael J. Barlage, Wei Yu, and Michael G. Duda. 2017. THE WEATHER RESEARCH AND FORECASTING MODEL Overview, System Efforts, and Future Directions. *BULLETIN OF THE AMERICAN METEOROLOGICAL SOCIETY* 98, 8 (AUG 2017), 1717–1737. https://doi.org/{10.1175/BAMS-D-15-00308.1}

[13] W. C. Skamarock, J. B. Klemp, J. Dudhia, D. O. Gill, M. Barker, K. G. Duda, X. Y Huang, W. Wang, and J. G. Powers. 2008. *A description of the Advanced Research WRF Version 3.* Technical Report. National Center for Atmospheric Research. 1–113 pages.

[14] Jon Whittle, Jeffrey Van Baalen, Johann Schumann, Peter Robinson, Thomas Pressburger, John Penix, Phil Oh, Michael Lowry, and Guillaume Brat. 2001. Amphion/NAV: Deductive synthesis of state estimation software. In *Automated Software Engineering, 2001.(ASE 2001). Proceedings. 16th Annual International Conference on*. IEEE, 395–399.