



Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Ausarbeitung

Interview Luis Kornblueh/ICON

vorgelegt von

Fin Töter

Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik
Arbeitsbereich Wissenschaftliches Rechnen

Studiengang: Software System Entwicklung
Matrikelnummer: 6995113
Betreuer: Dr. Hermann Lenhart

Hamburg, August 24, 2018

Contents

1	Das MPI	3
1.1	Geschichte MPI	3
1.2	Die Sclab Gruppe	3
2	ICON	5
2.1	ICON-ESM	5
2.1.1	Einführung	5
2.1.2	ICON-Auflösung	6
2.1.3	ICON Usage	7
3	Softwareentwicklung am MPI	8
3.1	Luis Kornblueh	8
3.2	Tools	8
3.2.1	Software	8
3.2.2	Programmiersprachen	9
3.2.3	Hardware	10
3.3	Klassisch vs. Wissenschaftlich	11
3.4	Nicht funktionelle Anforderungen	13
3.4.1	Qualitätsmanagment	13
3.4.2	Testing	13
3.4.3	Benchmarking	14
3.4.4	Datenmanagement	14
3.5	Bottlenecks	14
4	Kollaboration	15
4.1	Intern	15
4.1.1	Innerhalb der Gruppe	15
4.1.2	IT-Unterstützung	15
4.2	Extern	16
4.2.1	Deutscher Wetterdienst	16
4.2.2	DKRZ	16
4.2.3	Workshops/Tagungen	16
4.3	Zukunft Modellentwicklung	17

1 Das MPI

In diesem Kapitel werde ich zum einen das MPI als den Arbeitgeber von Luis vorstellen, sowie eine grobe Übersicht über die Struktur des MPI-M geben.

1.1 Geschichte MPI

Im Jahre 1974 wurde das Max-Planck-Institut für Meteorologie in Hamburg, mit der Aufgabe sich den aktuellen Fragestellungen auf dem Gebiet der langfristigen Klimaveränderungen zuzuwenden, gegründet[4], da diese im bisherigen Ausmaße in der Bundesrepublik Deutschland noch nicht aufgegriffen werden konnten. [4].

Zum Zeitpunkt der Gründung wurden auch bereits zwei der drei jetzigen Arbeitsbereiche, "Athmosphäre im Erdsystem" und "Ozean im Erdsystem" festgelegt: "Das neue Max-Planck-Institut für Meteorologie wird sich vornehmlich der Entwicklung von Zirkulationsmodellen der Atmosphäre und der Erforschung der Wechselwirkungen zwischen Ozean und unterer und oberer Atmosphäre widmen" [4]. Im Laufe der Jahre gesellte sich noch die Arbeitsgruppe "Land im Erdsystem" dazu.

Aktuell ist das MPI-M weltweit für eine Forschungen im Rahmen der Klimaforschung bekannt, dies macht sich auch bemerkbar dadurch, dass vier Forscher des MPI am IPCC 2020/21 mitwirken[1] und auch bei vorherhigen IPCCs mitgearbeitet wurde.

1.2 Die ScLab Gruppe

Die ScLab Gruppe oder Scientific Computing Lab ist eine von den drei Hauptarbeitsbereichen unabhängige Arbeitsgruppe, welche hauptsächlich zur Unterstützung von OES, AES und LES dient. Sie ist wiederum in drei verschiedene Teams aufgeteilt. Diese sind:

- CIS - Central IT Services

Die Gruppe kümmert sich um die allgemeine IT. Hierzu gehört, dass Software und Hardware Probleme gelöst werden, sowie der Betrieb der Server Infrastruktur und anderer IT Systeme.

- CIMD - Computational Infrastructe and Model Development

Diese Gruppe beschäftigt sich hauptsächlich mit der Entwicklung des Klimamodells ICON. Hierbei liegt der Fokus vor allem auf der Optimierung der von den Wissenschaftlern designten Modelle. Sowie auf dem Benchmarking und Testen dieser Modelle. In diesem Bereich arbeitet auch Luis.

- SIP - Strategic IT Partnerships
Eine Gruppe die sich hauptsächlich um, wie der Name schon sagt, strategische Partnerschaften mit anderen Instituten oder Firmen kümmert z.B. **DKRZ**, **DWD**, **AWI** und **KIT**

2 ICON

In diesem Kapitel werde ich versuchen das Modell, welches Luis entwickelt, ein wenig zu umreißen.

2.1 ICON-ESM

2.1.1 Einführung

Das ICON-ESM oder auch ICON Earth System Model, ist ein numerisches Wettervorhersagemodell, welches vom Deutschen Wetterdienst gemeinsam mit dem Max Planck Institut für Meteorologie entwickelt wird. Dieses Modell ist eines der beiden Hauptprojekte an denen die Arbeitsgruppe von Luis forscht.

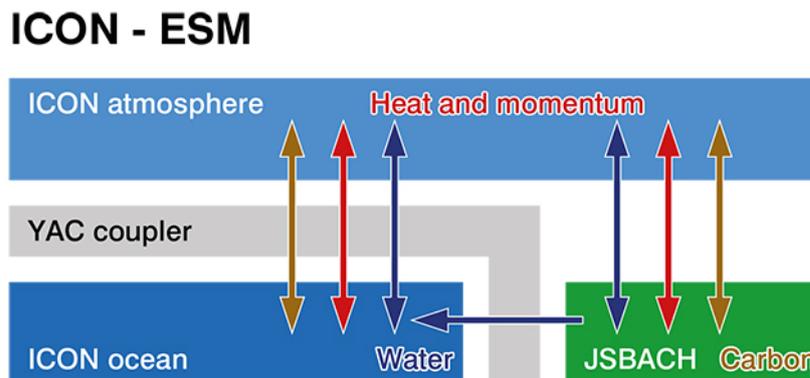


Figure 2.1: ICON Funktionsweise[2]

Wie in Figure 2.1 zu sehen, besteht ICON-ESM aus drei diskreten Teilen, sowie einem Koppler zwischen den einzelnen Teilen. Diese lassen sich genau auf die Abteilungen im MPI-M aufteilen.

- ICON Atmosphere → Atmosphäre im Erdsystem
ICON-A ist der Teil von ICON, der zuständig für die Simulation der Atmosphäre und somit verantwortlich dafür Luftbewegungen und Gase richtig zu berechnen.
- ICON Ocean → Ocean im Erdsystem
ICON-O ist der Teil von ICON der zuständig ist für die Simulation der Ozeane. Hierzu gehören Strömungen, Strudel, Erosion der Landmassen und weiteres.

- JSBACH → Land im Erdsystem
JSBACH ist der Teil von ICON, der für die Simulation der Landmassen zuständig ist. Hierzu gehören vor allem der Hitzetransport sowie der Wassertransport in der Erde.

Dabei ist aus der Figure 2.1 auch zu sehen, dass ICON ein Multiphysikmodell ist. Dies bedeutet das nicht nur eine physikalische Komponente simuliert wird, sondern jegliche physikalische Komponente, von Wasserströmung bis hin zu Übertragung von Gasen.

2.1.2 ICON-Auflösung

Eine weitere Besonderheit von ICON-ESM ist, dass es möglich ist, verschiedene Simulationsauflösungen in den verschiedenen Teilen von ICON einzustellen.

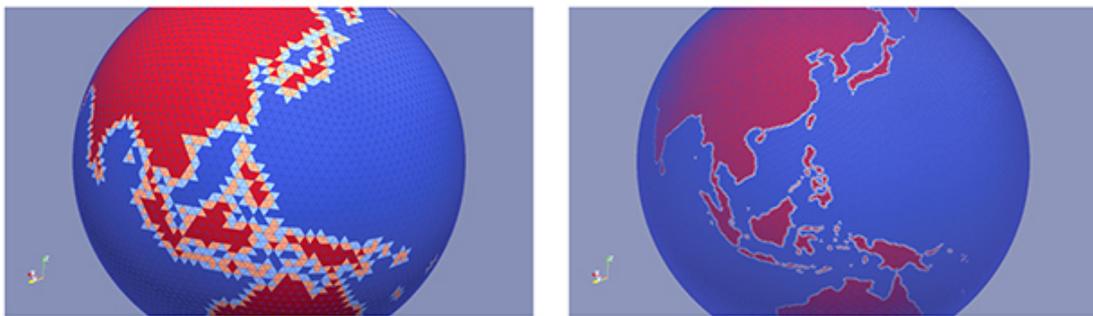


Figure 2.2: Typische Grid Konfiguration für die indonesische Region. Hierbei Auflösung von 60 km ICON-A (links) und 40 km ICON-O (rechts)[3]

In Figure 2.2 können wir eine typische Konfiguration von ICON betrachten. Hierbei ist vor allem zu beachten, dass bei niedrigeren Auflösungen durchaus Probleme auftreten können. Diese können wir in unserem Beispiel vor allem im Westen von Indonesien betrachten. Hier erkennen wir, dass bei zu kleiner Auflösung manche Wassermassen überhaupt nicht erkannt werden. Dies wäre zum Beispiel auch der Fall, wenn wir England und den Golfstrom simulieren wollen, da bei zu grober Auflösung der Kanal zwischen England und Frankreich nicht mehr simuliert wird. Hieraus sollte klar werden, dass es von großer Bedeutung ist, Auflösungen variabel setzen zu können. Nicht nur in Bezug auf die Optimierung sondern auch auf die Exaktheit der Simulation.

YAC-Coupler

Ein weiterer wichtiger Teil von ICON ist der YAC Coupler. Dieser wurde am **DKRZ** verwendet und verbindet ICON-A mit ICON-O. Er ist dafür zuständig, dass die Werte richtig übergeben werden und eine Skalenanpassung anwenden. Dies ist vor allem für die Funktion der dynamischen Auflösung relevant. Er ist zum Beispiel zuständig für den Austausch von :

- Energie
- Bewegung
- Wasser
- Wichtigen Gasen

2.1.3 ICON Usage

Wir haben bisher viel darüber gelernt was ICON ist und wie man es konfigurieren kann. Doch was macht ein Wissenschaftler jetzt eigentlich mit so einem Modell? Im Allgemeinen gibt es vor allem zwei verschiedene Arten ein Modell zu verwenden.

- Validation
- Projektion

Diese beiden Arten kann man in zwei Phasen aufteilen:

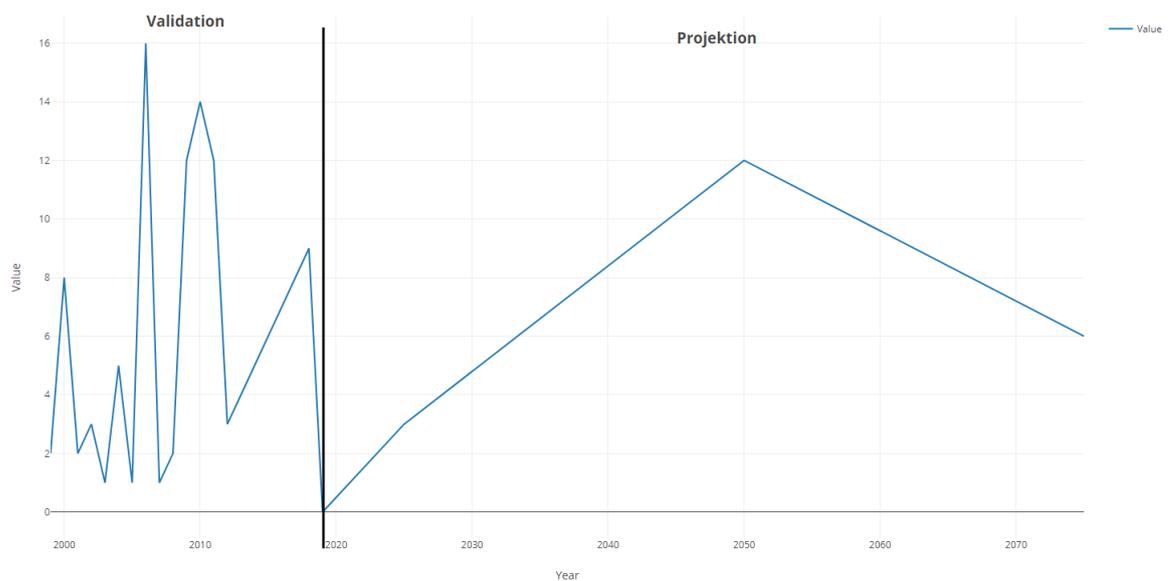


Figure 2.3: Validation und Projektionsphasen

In der Validationsphase wird versucht mit dem Modell Werte zu validieren, welche man vorher gemessen hat. Hierzu simuliert man beispielsweise den Temperaturverlauf vom 1.1-1.2 2017 in Hamburg und vergleicht diesen mit den wirklich gemessenen Werten. Wenn man dies häufig genug mit genügend Variablen getestet hat, kann man davon ausgehen, dass das Modell validiert ist und kann nun in der Projektionsphase zukünftige Ereignisse und Zeitpunkte simulieren.

Eine weitere Art der Verwendung ist das Experimentieren. Beispielsweise lässt sich mit ICON auch eine komplette Eisswelt simulieren oder man simuliert die Folgen eines Ausfallens der Sonnenstrahlung.

3 Softwareentwicklung am MPI

Hier werden ich genauer über das Interview und über die Antworten, die Luis mir gab, berichten.

3.1 Luis Kornblueh

Bevor wir zu den wirklichen Interviewfragen kommen, sollten wir erst einmal klären wer Luis Kornblueh überhaupt ist:

Luis Kornblueh ist ein wissenschaftlicher Programmierer am MPI-M, welcher hauptsächlich an ICON-ESM arbeitet. Luis selber hat in seiner Laufbahn mit einem Meteorologiestudium an der TH Darmstadt angefangen, in dem er bereits einen großen Fokus auf Optimierung von Modellen legte. Nachfolgend arbeitete er bereits am Forschungszentrum Geesthacht im Rahmen der Modellentwicklung. Er wechselte dann jedoch nach einiger Zeit in die Entwicklung von Algorithmen zur Auswertung von Satellitendaten und legte hier vor allem einen Fokus auf die Entwicklung nach Fortran Standard. Darauf folgend führte ihn sein Weg zur Universität Hamburg wo er im Jahr 1993 seine Promotion im Bereich der Modellentwicklung begann. Nach Abschluss seiner Promotion im Januar 1997 fing Luis im März desselben Jahres am MPI mit dem Fokus auf der Modellentwicklung an.

Auf die Frage was ihn denn an seinem Job begeistert, antwortete er , dass es ihn begeistert andauernd neue Probleme zu lösen, die schließlich 90% interessant sind und nur 10% ätzend seien.

3.2 Tools

Am Anfang des Gesprächs fragte ich Luis nach den Tools, welche er in seinem normalen Arbeitsalltag verwendet. Diese teilen sich ein in Software, Hardware und Programmiersprachen.

3.2.1 Software

- Numerische Modelle
- Datenformate: netcdf, hdf5, crypt2
- Datenbanken: PostgresSQL

- Tools zum verteilten Zugriff auf Datenbanken
- OpenMPI
- kryptographische Tools
- Compiler
- Shells etc.

Wenn man einen Blick auf diese Liste wirft, fällt einem nichts ungewöhnliches bei auf. Hinsichtlich der klassische Programmierung scheint es hier keinerlei großartige Unterschiede zu geben. Man sieht allerdings bereits, dass Luis einen sehr großen Wert auf Optimierung legt. Zum Beispiel scheint er auf verteilten System (OpenMPI) zu arbeiten und auch einen großen Wert auf optimale Dateiformate zu legen(hdf5,crypt2).

3.2.2 Programmiersprachen

- C
- Fortran
- C++
- Python
- Bash
- Assembler

Hier sagt schon der erste Blick auf die Liste aus, was der Mensch dahinter macht. C, Fortran und Python sind immernoch die drei größten HPC sprachen. Da vor allem in einer Simulation sehr optimal programmiert werden muss, um die höchstmöglichen Years/Day(Die Anzahl der simulierten Jahr die an einem echten Tag simuliert werden können) zu erreichen.

Auch ICON zeigt eine derartige Verteilung der Sprachen:

- mehrere 100k Zeilen C
- mehrere 100k Zeilen Fortran
- ein wenig Assembler

Das einzige was mich bei dieser Verteilung überraschte war, dass Luis tatsächlich auch Assembler selber schreibt und sich nicht nur von Compilern generieren lässt.

3.2.3 Hardware

Der Hardwaregebrauch von Luis:

- 99,98% Verwendung Mistral
- 0,02% Verwendung Bürorechner
- 100% Verwendung **CPU** keine GPUs → Mistral pure CPU Maschine

der vor allem interessante Punkt des oben beschriebenen Hardwaregebrauchs ist die ausschließliche Nutzung von CPUs für die Berechnung der Simulation. Hier wurde ich auch stutzig, schließlich hatte ich im Informatik Studium gelernt, dass GPU's bei solchen Berechnungen schneller wären. Hierbei erzählte mir dann mein Betreuer die Geschichte, dass Luis einst bei einer Vorstellung von NVIDIA gefragte wurde, ob es sich nicht lohnen würde, auch GPUs für ICON einzusetzen. Dies verneinte Luis, da es alleine 1 Jahr dauern würde, die Codebase auf GPUs umzustellen. Dies ist sehr interessant, da der Trend im HPC eigentlich eher Richtung GPU Berechnungen geht, wie auch an der Liste der TOP500 erkennbar wird:

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	Summit - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , IBM DOE/SC/Oak Ridge National Laboratory United States	2,282,544	122,300.0	187,659.3	8,806
2	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , NRCP National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
3	Sierra - IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband , IBM DOE/NNSA/LLNL United States	1,572,480	71,610.0	119,193.6	
4	Tianhe-2A - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000 , NUDT National Super Computer Center in Guangzhou China	4,981,760	61,444.5	100,678.7	18,482
5	AI Bridging Cloud Infrastructure (ABCI) - PRIMERGY CX2550 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR , Fujitsu National Institute of Advanced Industrial Science and Technology (AIST) Japan	391,680	19,880.0	32,576.6	1,649

Figure 3.1: Top 5 Supercomputers[6]

Hier haben schließlich $\frac{4}{5}$ der Top 5 schon Graphikkarten. und das Verhältnis zieht sich größtenteils durch die Liste.

3.3 Klassisch vs. Wissenschaftlich

Die Hauptfrage, die ich Luis stellte, war, was seiner Meinung nach die Definition von Softwareentwicklung sei. Er konnte mir zwar keine genaue Definition geben, jedoch gab er mir drei Stichpunkte, welche nach seiner Meinung die wichtigsten Komponenten der Softwareentwicklung sind:

- Planung
- Kommunikation
- Validation

Außerdem machte er mir mit Nachdruck klar, dass **Programmieren kein Teil der Softwareentwicklung sei**, sondern nur ein Werkzeug. Diese Aussage ging exakt gegen das, was mir im Studium gelehrt wurde. Im Studium wurde uns gelehrt, dass Softwareentwicklung das Erstellen und Programmieren von möglichst eleganter Software sei. Dabei sollten vor allem gemäß den ISO 9126 die Qualitätsmerkmale Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Wartungsfreundlichkeit und Übertragbarkeit im Vordergrund stehen. Und, dass Planung und Kommunikation nur ein **Werkzeug** seien, um dies zu erreichen. Dies ist auf jeden Fall eine große Diskrepanz zwischen dem, was uns gelehrt wird und was ein wissenschaftlicher Programmierer wie Luis unter SE versteht.

Im Laufe des Gesprächs kamen weitere Diskrepanzen zum Vorschein.

- Subroutinen für Strukturierung nicht für Wiederverwertung
Luis erzählte mir, dass er meistens in seinen Projekten Subroutinen zur Strukturierung verwendet. Hierbei wurde ich stutzig, da mir im Studium beigebracht wurde Subroutinen zwar auch zur Strukturierung zu verwenden sie, jedoch hauptsächlich auf die Wiederverwertung auszulegen. Immer mit dem Gedanken im Kopf "Könnte ich diese Funktionalität noch irgendwo anders gebrauchen?".
- Software-Life-Cycle
Ich stellte Luis während unseren Gesprächs außerdem die Frage, welchen Life-Cycle ICON hätte und ob es irgendwann ein ICON 2 oder etwas in der Art geben würde. Darauf lachte er nur und erzählte mir, dass sie inkrementell ICON immer weiter entwickeln und somit ein komplett neuer Aufbau nicht geplant ist. Auch hierbei war ich verwirrt, da ich im Studium gelernt habe, dass es durchaus sinnvoll ist, alle paar Jahre mal von nahezu Grund auf neu anzufangen, da ansonsten Systeme "overbloated" werden. Weiterhin veralten Systeme durch die schnelle Entwicklung der Informatik sehr schnell und es gibt schneller oder flexiblere Wege Probleme zu lösen. Wobei ich ehrlich gesagt nicht sicher bin, ob sich so etwas auf Klimamodelle anwenden lässt.
- Assembler für konstante Geschwindigkeit und nicht für schnelleren Code
Zu Anfang dieser Ausarbeitung habe ich erwähnt, dass Luis tatsächlich auch manuell

Assembler schreibt. Dies hielt ich für sehr ineffizient, da ich vermutete, dass GCC oder andere Compiler bei weitem besser kompilieren als ein Programmierer es per Hand tun könnte. Darauf wurde mir erklärt, dass in diesem Fall Luis den Assembler tatsächlich deshalb selber schreibt, auf Grund der Tatsache, dass er dann genau weiß wie viele Zyklen der Code braucht, um ausgeführt zu werden. Im Vortrag wurde gefragt warum dies denn wichtig sei. Auf Nachfrage antwortete Luis, dass er selber Assembler hauptsächlich schreiben würde um verschiedene Compiler zu vergleichen und sie auf ihre Geschwindigkeit zu überprüfen. Bei GCC zum Beispiel ist diese Ausführzeit je nach System ,je nach Compiler flags und je nach Compiler Version anders.

3.4 Nicht funktionelle Anforderungen

In meinem Gespräch mit Luis haben wir natürlich auch über nicht funktionale Anforderungen gesprochen wie:

- Qualitätsmanagment
- Testing Benchmarking

diese werde ich nun vorstellen.

3.4.1 Qualitätsmanagment

Mit der Art von Qualitätsmanagment, die ich im Studium gelernt habe, hat Luis weniger zu tun. Beispielsweise lernte ich in dem Modul Softwaretechnik die ISO/IEC 9126 kennen, die Dinge regeln wie Usability, Responsibility etc. Von diesen Dingen hatte Luis zwar schon einmal gehört, jedoch werden sie in ICON nicht explizit durchgesetzt. Zwar vermute ich, dass sie implizit schon verwendet werden, da schließlich auf Usability etc. geachtet wird, allerdings nicht auf Grund der ISOs.

Ein wichtiger Punkt sind jedoch die Programmierkonventionen, die in ICON strikt verwendet werden. Es gibt ein Stylesheet, welches Konventionen für C und Fortran und weiteres festlegt.

Der dritte Punkt, auf den ich ihn ansprach war, ob es denn irgendwelche Restriktionen hinsichtlich von Komplexität gebe. Dies bestätigte er und erzählte mir, dass im Quality-of-Life Teil des Projektes durchaus $O(n^4)$ verwendet werden darf. Im Kern gelten jedoch sehr viel striktere Regeln. Außerdem sei auch festgelegt, welche Datenstrukturen benutzt werden dürfen und vieles weiteres.

3.4.2 Testing

Der Bereich des Testings, erzählte Luis, sei sehr wichtig. Diese Aussage deckte sich weitestgehend mit dem, was ich aus dem Studium gelernt habe. Die Tests teilen sich in hauptsächlich zwei Teile auf einmal die semantischen Tests und die syntaktischen Tests: Die syntaktischen Tests bestehen bei Luis in der Gruppe einmal aus Continuous Integration Tests, sowie einem build-bot, welcher dauerhaft versucht, ICON zu kompilieren und Fehler auswirft, wenn diese beim kompilieren entstehen.

Auf der andere Seite sind die semantischen Tests, welche bei einem Projekt wie ICON sehr viel größer werden, als ich es bei von Studienprojekten gewöhnt war. Für die semantischen Tests werden monatelang Modelle laufen gelassen um die Ergebnisse dann automatisch und teils auch manuell zu verifizieren. Hierbei ist natürlich eine simulierte Zeit gemeint, genauso wie bei jeder nachfolgenden Zeiteinheit. Weiterhin werden bei jedem Release noch einmal die einzelnen Teile getestet. Dies geschieht mit Hilfe von:

- AMIP (Athmosphere Model Intercomparison Project) Lauf → 2-3 Tage
- OMIP (Ocean Model Intercomparison Project) Lauf → 2 Tage

- CMIP (Coupled Model Intercomparison Project) Lauf → 6 Wochen

3.4.3 Benchmarking

Benchmarking ist ein weiterer sehr wichtiger Teil in Luis Arbeit. Hier möchte er vor allem sehen, wie seine Modelle sich auf der Mistral verhalten und wie lange die einzelnen Programmabschnitte benötigen, um diese dann weitergehend zu optimieren. Hierzu wurde eine hausinterne Softwarebibliothek entwickelt, welche es ermöglicht, Timer in das Programm einzubauen und die Zeit von Abschnitten zu messen. Die Daten dieser Timer lassen sich dann in HDF5 exportieren und in einem von mir entwickelten Webfrontend anzeigen lassen.

3.4.4 Datenmanagement

Die Frage, die in einer Gruppe in einem Studienprojekt immer zuerst aufkommt, ist: "Benutzen wir GIT?". Diese Frage wird natürlich meistens positiv beantwortet und es scheint, dass sich dieser Trend in der Wissenschaft weiter durchzieht. Luis und seine Gruppe verwenden GIT für die Versionisierung ihres Quellcodes.

Außerdem ist im Bereich des HPCs auch die Größe der Daten ein relevanter Faktor, mit dem man rechnen muss. Luis erzählte mir, dass sie teils nur 100GB an Eingabedaten hätten, am Ende jedoch mehr als 120 PB an Ausgabe produzieren. Dies ist fast das 1,2 millionenfache. Diese sagte Luis, würden aber im Nachhinein auf ca. 1 PB echte Daten zurück geschrumpft. Weitergehend werden die Daten dann ins **DKRZ**-Archiv verschoben und hier jahrlange gelagert, da Daten von wissenschaftlichen Arbeiten oder IPCC Berichten rechtmäßig sehr lange gelagert werden müssen.

3.5 Bottlenecks

Eine der für mich interessantesten Fragen war die Frage nach Bottlenecks in Luis Workflow. Bei dieser Frage hatte ich eigentlich als Antwort erwartet, dass Luis Dinge nennt wie, I/O Performance oder Rechenpower. Aber überraschenderweise antwortete Luis sehr simpel mit der Aussage, dass ihm einfach Mitarbeiter fehlen. **Manpower** sei sein größtes Bottleneck. Er hat viele Dinge, die er implementieren möchte, jedoch nicht genügend Mitarbeiter, um diese Menge zu schaffen.

4 Kollaboration

In diesem Kapitel möchte ich ein wenig über die internen und externen Kollaborationen sprechen, die Luis erwähnt hat, sowie einmal darlegen, welche Themen Luis auf Workshops oder Tagungen vorstellt.

4.1 Intern

Im Rahmen der internen Kollaboration sind zwei Dinge wichtig: Einmal die Zusammenarbeit innerhalb der Gruppe sowie die Unterstützung der IT, die in einem Job, der sehr viel mit Programmierung zu tun hat, durchaus relevant ist.

4.1.1 Innerhalb der Gruppe

Intern erfolgt die Kollaboration zu meist über Tür zu Tür Kommunikation, jedoch auch die Standardkommunikationsmittel wie Telefonate und ab und zu ein größeres Meeting. Allerdings werden auch agile Softwareentwicklungsmethoden verwendet wie zum Beispiel Kanban.

Dies heißt, dass sich die gesamte Gruppe jeden Tag um eine bestimmte Uhrzeit trifft und jeder erzählt, welche Dinge er gestern getan hat, sowie die Dinge, die er heute tun möchte. Weiterhin gibt es ein Kanban Board, an dem verschiedene Projektaufgaben priorisiert hängen.

4.1.2 IT-Unterstützung

Die IT-Unterstützung wird wie oben schon gesagt am MPI vom Central IT Service übernommen. Luis Meinung nach liefert die CIS-Gruppe ein gutes Ergebnis ab, jedoch würde die Installation von neuer Software sehr langsam ablaufen. Da dies für Luis, der sehr gerne neue Software testet und versucht in ICON einzubauen, ein größeres Problem darstellt, hat er sich ein ca. 500 Zeilen Bash Script geschrieben, welches seine Standardsoftware und Pfade installiert, wodurch er neue Dinge sehr schnell bekommt.

4.2 Extern

Die externe Kollaboration ist zu meist mit dem **Deutschen Wetterdienst**, im Nachfolgenden als **DWD** bezeichnet, in Offenbach in Hinsicht auf die Entwicklung von ICON und hinsichtlich der Datenaufbereitung und Datenarchivierung mit dem **DKRZ**. Weiterhin stellt er seine Erkenntnisse auf Workshops und Tagungen vor.

4.2.1 Deutscher Wetterdienst

Eine der beiden engen externen Zusammenarbeiten ist der Deutsche Wetterdienst. Dieser ist zusammen mit dem MPI-M hauptverantwortlich für die Entwicklung von ICON. Das bedeutet, hier findet zweimal jährlich ein Entwicklertreffen beim **DWD** statt, weiterhin gibt es regelmäßige Videokonferenzen, auf den die wichtigsten Dinge und Neuerungen besprochen werden. Außerdem wird hier die Slack Technologie angewandt und man vereinbart gemeinsame Sprint Sessions. Natürlich werden auch normale Kommunikationskanäle wie Telefon, E-Mail und weiteres verwendet.

4.2.2 DKRZ

Das **DKRZ** ist vermutlich der Hauptverantwortliche wenn es um die Datensätze geht. Hierbei sprach Luis davon, dass die Unterstützung des **DKRZ** vor allem auf persönlicher Ebene sehr freundlich und angenehm ist. Auch die Arbeit auf Mistral wäre sehr einfach, da I/O Operationen wie gewohnt mit Unix Operationen wie "mkdir" oder "cp" gemacht werden können. Allerdings gäbe es auch hier dieselben Probleme wie bei der internen IT Unterstützung, das Nachinstallieren von Dingen würde sehr lange dauern. Jedoch verwendet Luis hier einfach dasselbe Skript wie auch schon im internen Bereich, was das ganze wieder ein wenig relativiert. Im Großen und Ganzen empfindet Luis die Zusammenarbeit mit dem **DKRZ** als sehr gut.

4.2.3 Workshops/Tagungen

Da Luis stark im Wissenschaftlichen Bereich tätig ist, fragte ich ihn, was er auf Workshops oder Tagungen, wie z.B. IS-INES oder ESI-WAGS erzählen würde, da er schließlich hauptsächlich im internen Code von ICON arbeiten würde und hierbei natürlich nichts gezeigt werden kann. Darauf antwortete Luis mit folgenden Dingen:

- Kompression von Daten
- Vorteile von Formaten
- Vorteile von Compilern
- Gedanken über I/O Subsysteme
- Parallelität in Programmen erklären

- Optimierungen auf verschiedenen Ebenen

Diese Liste repräsentiert wunderbar die Interessen und Arbeitsbereiche von Luis. Hierbei geht es vor allem um Optimierung von Dingen, dabei ist es ihm egal, auf welcher Ebene diese Optimierungen stattfinden. Diese können auf Filesystemebene (Formate,I/O Subsysteme) oder auf Programmierenebene(Parallelität) oder auf noch anderen Ebenen sein, Luis würde sich dafür interessieren.

4.3 Zukunft Modellentwicklung

Die abschließende Frage über die Luis und ich sprachen, war die Aufgeschlossenheit Luis' gegenüber Neuerungen in der Informatik, und welche Trends er in der Modellentwicklung sieht. Hierbei sprach ich ihn zuerst auf das Thema Berechnungen in der Cloud an, da auch die Hersteller von Formaten wie HDF5 inzwischen Cloud Dienste anbieten zum Beispiel(HDFCloud [5]). Diese Frage verneinte Luis, da seiner Meinung nach durch die Größe der Datenmengen die Cloud einfach preislich nicht kompetitiv wäre.

Ein weitere Punkt den er ansprach, war, dass er Neuerungen eher in der eigenen Entwicklung sieht, da ein großes Gap zwischen dem was sich Informatiker ausdenken und dem was Wissenschaftler benötigen, existieren würde. Dadurch seien dann Neuerungen aus der Informatik für ihn weniger relevant.

Ein Trend an der in der Informatik aktuell mit großen Interessen erforscht wird, ist der des Machine Learnings. Auch hierauf sprach ich Luis an, doch er erzählte mir, dass aktuell die Skalierung von Machine Learning Algorithmen hinsichtlich der Datenmengen noch nicht sonderlich gut sei.

Bibliography

- [1] Dörte de Graaf. *IPCC benennt neue Autoren: Hamburger Forscher übernehmen wichtige Funktionen beim sechsten IPCC-Bericht*. [Online; accessed 24 Octobre, 2018]. 2018. URL: <https://www.mpimet.mpg.de/kommunikation/aktuelles/single-news/news/ipcc-benennt-neue-autoren-hamburger-forscher-uebernehmen-wichtige-funktionen-beim-sechsten-ipcc-beri/>.
- [2] Max Planck Institut für Meteorologie. *ICON Development*. [Online; accessed June 27, 2018]. ? URL: <https://www.mpimet.mpg.de/en/communication/news/focus-on-overview/icon-development/>.
- [3] Max Planck Institut für Meteorologie. *ICON Earth System Model*. [Online; accessed June 27, 2018]. ? URL: <https://www.mpimet.mpg.de/en/communication/news/focus-on-overview/icon-earth-system-model/>.
- [4] MPG. *Institutsgeschichte Meteorologie*. http://pubman.mpd1.mpg.de/pubman/item/escidoc:2375806:4/component/escidoc:2375805/MPG_Handbuch_zur_Institutsgeschichte_-Meteorologie.pdf. 2013.
- [5] The HDF Group. *Hierarchical Data Format, version 5*. <https://www.hdfgroup.org/wp-content/uploads/2017/10/cloud-architecture-image.png>. 1997-2018.
- [6] Top 500 ORG. *Top 500 Hune 2018*. [Online; accessed Juli 9, 2018]. ? URL: <https://www.top500.org/lists/2018/06/>.