CiTAR – Citable Scientific Software and Software Methods

Klaus Rechert, Rafael Gieschke, Susanne Mocken, Oleg Stobbe, Oleg Zharkov[1]
Felix Bartusch[2] , Kyryll Udod[3]

[1]Universität Freiburg
[2]Universität Tübingen
[3]Universität Ulm

Bewahrung virtueller Forschungsumgebungen: CiTAR

Moderne Forschungsvorhaben stützen sich nicht nur auf computergestützte Methoden und digitale Ressourcen, sondern entwickeln häufig auch spezifische Software und software-gestützte Prozesse. Ein wesentliche Säule der Wissenschaft ist die Nachvollziehbarkeit sowie gegebenenfalls die Reproduktion oder Verifikation veröffentlichter Erkenntnisse. Für computergestützte Forschung wird dies zunehmend zu einer Herausforderung, insbesondere durch die komplexe Rekonstruktion software-basierter Methoden, ihrer spezifischen Konfiguration und technischen Abhängigkeiten. Dieser Beitrag stellt CiTAR[1] - *Citing and Archiving Research* vor, ein dreijähriges Projekt gefördert durch das Ministeriums für Wissenschaft und Kunst des Landes Baden-Württemberg, das eine Infrastruktur zur Unterstützung computergestützter Forschung entwickelt. Im Fokus stehen dabei die Methoden, d.h. softwaregestützte Prozesse oder Modelle zur Erstellung oder Auswertung von Daten. Das wesentliche Projektziel ist es, diese Methoden nachvollziehbar und nachnutzbar nachzuweisen, so dass diese ebenso zitierbar und publizierbar werden wie derzeit schon Forschungsdaten (vgl. z.B. Brase 2009).

Sowohl die Erhaltung als auch die Zitierfähigkeit von (wissenschaftlicher) Software als weiterer grundlegender Aspekt einer Forschungsdatenstrategie ist im wesentlichen unbestritten[2] [3], diverse Akteure und Organisationen haben sich bereits dem Thema bereits gewidmet.[4] [5] Für eine erfolgreiche Reproduktion oder Verifikation (z.B. definierbar als "*the ability for someone to replicate a computational experiment that*

[1] http://citar.eaas.uni-freiburg.de/
[2] National Institutes of Health (NIH), https://grants.nih.gov/grants/guide/notice-files/NOT-OD-17-015.html
[3] https://danielskatzblog.wordpress.com/2017/02/22/creation-publication-and-citation-issues-in-software- citation-versus-paper-and-data-citation/
[4] Software Preservation Network http://www.softwarepreservationnetwork.org
[5] UNESCO PERSIST Initiative, https://www.unesco.nl/digital-sustainability

was done by someone else, using the same software and data, and then to be able to change part of it (the software and/or the data) to better understand the experiment and its bounds[.]"[6]) ist das Vorhalten von einzelnen Softwarekomponenten nicht ausreichend. Software-gestützte Prozesse bestehen meist aus mehreren Softwarekomponenten, die einer spezifischen Konfiguration und Orchestrierung bedürfen. Für zitierfähige und nachnutzbare Softwaremethoden sind also eigenständig nutzbare Einheiten notwendig, die eine Kapselung spezifischer Softwareinstallationen ermöglichen. Im Rahmen von CiTAR wurden virtuelle Maschinen (VMs) und Container (Meng et al. 2015, Boettinger 2015) als geeignete Einheiten identifiziert und ein Prozess entwickelt, um diese als eigenständige Objekte aus den Labor- und Arbeitsumgebungen der Entwickler und Forschenden (Produzenten) herauszulösen und von diesen unabhängig funktional zu erhalten. Ein publiziertes Zitat impliziert langfristig aber auch Verfügbarkeit, so dass die Langzeiterhaltung der so nachgewiesenen Methoden eine wichtige Nebenbedingung für die im Projekt entwickelten Konzepte und Infrastruktur darstellen. Ein wesentliches Ziel war somit, für VMs und Container jeweils eine homogene, technische Objektdefinition zu entwickeln, so dass die Erhaltungsplanung sowie spätere Erhaltungsmaßnahmen -- mit nur wenigen verschiedenartigen Objektarten -- effizient erfolgen können.

Erhaltung und Nachnutzung virtueller Maschinen

Als Beispiel für die Aufrechterhaltung eines langfristigen Zugangs zu softwarebasierten Forschungsressourcen haben wir unter anderem die Ergebnisse des Projekts SlaVaComp (2013-2015) genutzt, das ein elektronisches Meta-Glossar mit regionalen und diachronen Varianten des Kirchenslavischen erstellt hat -- eine Sprache, die im orthodoxen slavischen Raum zwischen dem 10. und 16. Jahrhundert verwendet wurde. Bis zur Erstellung einer digitalen Datenbank hatten Forscher nur die Möglichkeit, gedruckte Wörterbücher zu konsultieren, so dass selbst einfache Recherchen sehr viel Zeit in Anspruch nehmen konnten. Fünfzehn gedruckte kirchenslavisch-griechische und griechisch-kirchenslavische Glossare mit verschiedenen Herkunftsregionen wurden zu einer einfach zu bedienenden, webbasierten Online-Anwendung zusammengefasst.

[6] https://danielskatzblog.wordpress.com/2017/02/07/is-software-reproducibility-possible-and-practical/

Die Kosten für die Wartung eines Servers und der Software nach Projektende werden in der Regel nicht durch den Mittelgeber übernommen, insbesondere wenn diese Projekte nur für eine kleine und spezialisierte Forschungsgemeinschaft von Interesse sind. Der Betrieb einer nicht gewarteten, veralteten Maschine, die mit dem Internet verbunden ist, stellt jedoch ein latentes und zunehmendes Sicherheitsrisiko dar. Da der Support für das zugrunde liegende Serverbetriebssystem in naher Zukunft ausläuft, ist die Zukunft des SlaVaComp-Dienstes ungewiss. Selbst wenn das Betriebssystem auf die nächste Serverversion mit Langzeitsupport aktualisiert wird, ist damit nicht sichergestellt, dass andere Softwareabhängigkeiten, wie z.B. die zugrundeliegende Datenbank, kompatibel bleiben oder Sicherheitsupdates langfristig zur Verfügung stehen werden. Dies ist jedoch für einen öffentlichen Online-Dienst entscheidend. Eine Migration des Services auf eine modernen Softwarebasis ist aufwendig und bedarf der Mithilfe der ursprünglichen Entwickler, die aber in der Regel nach dem Ende des Projekts nicht mehr da sind und somit das umfassende, spezifische Wissen über die von ihnen erstellte Software mitgenommen haben. Dieses Schicksal teilen zahlreiche Softwareentwicklungen, die aus wissenschaftlichen Projekten hervorgehen.

Publikation einer virtuellen Forschungsumgebung

Virtuelle Maschinen sind in der Lage, sogenannte virtuelle Forschungsumgebungen (VRE) zu kapseln, die dann interaktiv oder mittels technischer Schnittstellen (API) von Forschenden genutzt werden können. Beispiele hierfür sind beispielsweise (Web-)Services, Content Management Systeme, Datenbanken, Simulationen oder ähnliche Dienste, die im Rahmen von Forschungsprojekten entwickelt wurden. Da jeder Forschungsbereich unterschiedliche Software-Stacks, spezifische Konfigurationsmöglichkeiten und Nutzungsmuster verwendet, ist ein generischer und vollautomatischer Ansatz erforderlich, der im Idealfall alle virtuellen Maschinen (VMs) gleich behandelt. Als zugrunde liegende Erhaltungsstrategie wird Emulation (bzw. Virtualisierung) auf Basis des Emulation-as-a-Service (EaaS)-Frameworks mit einer entsprechenden Integrationsstrategie für generalisierte/normalisierte Disk-Images und VMs eingesetzt (vgl. Rechert et al. 2012, Rechert et al. 2016).

Im Rahmen dieses Prozesses kann ein CiTAR-Publisher ein Festplattenabbild (Disk-Image) einer VM hochladen, das anschließend analysiert und technisch normiert wird, um die Kompatibilität mit der Emulations- oder Virtualisierungsinfrastruktur und deren Langzeitarchivierungsoptionen sicherzustellen. Darüber hinaus muss der Publisher die Maschine und deren Einsatzmöglichkeiten so beschreiben, dass eine zukünftige Nachnutzung (z.B. Inbetriebnahme) einfach möglich ist (s. Abb 1.). Anschließend ist der Publisher in der Lage, die Maschine zu testen und ggf. weiter anzupassen, beispielsweise um die Benutzerfreundlichkeit zu verbessern, Passwörter zu ändern, Anwendungen automatisch zu starten etc. und dann abschließend die korrekte Funktionsweise abzunehmen. Alle Änderungen werden (technisch) dokumentiert und erzeugen versionierte Revisionen der publizierten VM. Schließlich wird der veröffentlichten Maschine eine persistente Kennung von *Handle.net* zugewiesen, die auf eine *Landingpage* der archivierten Maschine zeigt.

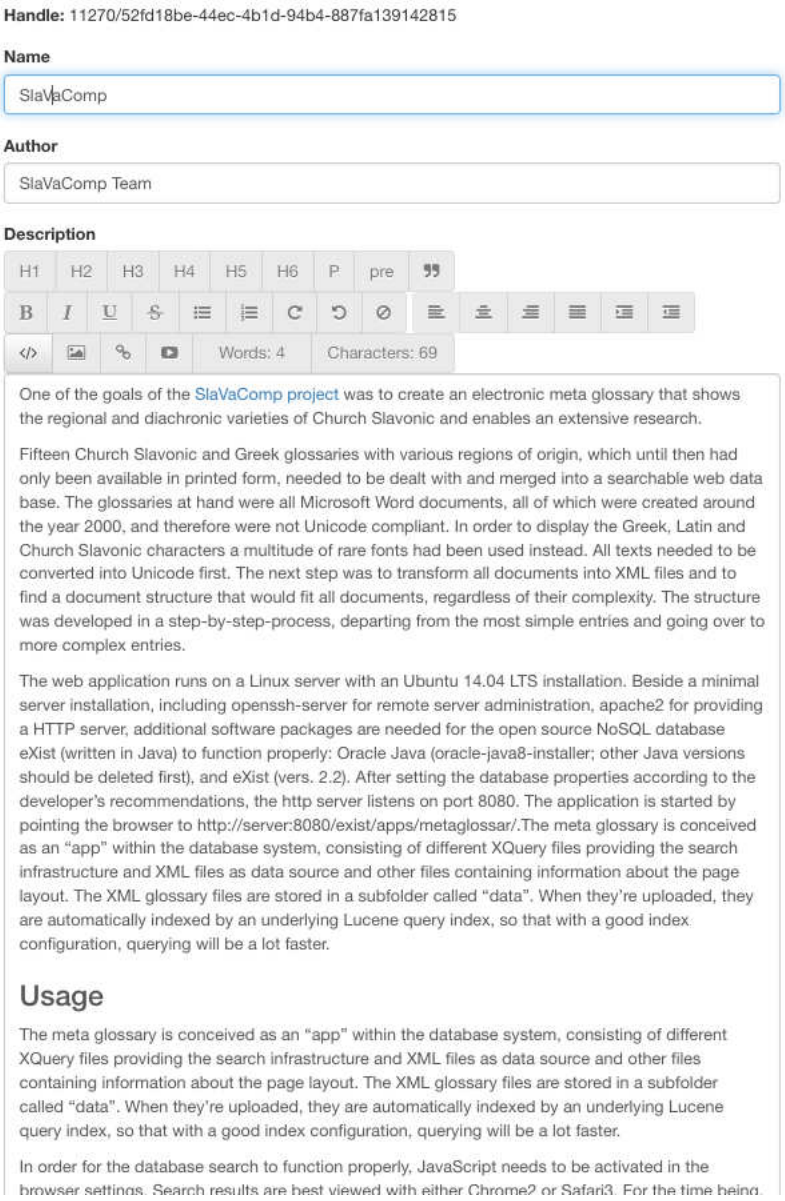

Abb. 1: Beschreibung einer Virtuellen Maschine.

Effizienter Betrieb und Zugriff

Eine veröffentlichte, archivierte und insbesondere zitierbare Forschungsumgebung erfordert auch eine langfristige technische Nutzbarkeit. Das *Handle*, das der archivierten VM zugeordnet ist, leitet den Benutzer zu einer *Landingpage* weiter, die

die virtuelle Umgebung und mögliche Interaktionen beschreibt. Die meisten dieser VMs richten sich an kleine Zielgruppen und werden nicht oft aufgerufen, so dass die VMs erst bei Bedarf gestartet werden.

Die größte Herausforderung betreffend des öffentlichen Zugriffs ist die Sicherheit der archivierten Maschine. Da ein solches System in seinem ursprünglichen Zustand archiviert wurde, ist die VM im Internet gefährdet. Um dennoch einen (sicheren) Netzwerkzugriff zu ermöglichen, werden alle archivierten Maschinen in einem eigenen privaten Netzwerk bereitgestellt. Die Brücke zwischen dem privaten Netzwerk und dem Netzwerk des Benutzers bildet das CiTAR-Gateway. Je nach Sicherheitsanforderungen und/oder Zugangsinfrastruktur stehen derzeit drei verschiedene Netzwerkzugangsoptionen zur Verfügung:

1. *Port forwarding* Für lokale Netzwerkimplementierungen können einzelne Netzwerkports eines emulierten Servers an ein dem Benutzer zugängliches Netzwerk weitergeleitet werden.

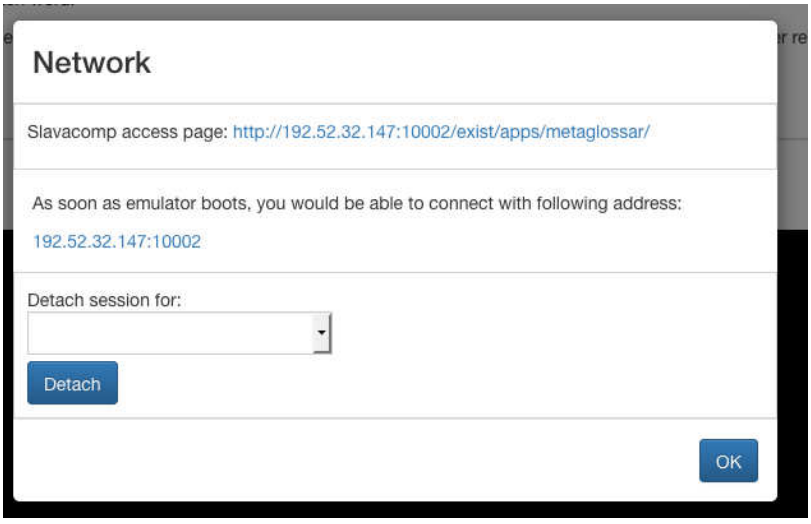

Abb. 2: Das CiTAR SlaVaComp Landingpage-Beispiel (hdl: 11270/52fd18be-44ec-4b1d-94b4-887fa139142815) zeigt ein Fenster mit Verbindungsinformationen an: Der Benutzer kann sich direkt mit der Webanwendung, die auf dem internen Port 8080 läuft, verbinden.

2. *SOCKS* Wenn die Maschine mehrere Dienste über TCP-Ports bereitstellt, kann der SOCKS5-Modus verwendet werden. Das Fenster “Verbindungsinformationen” zeigt die notwendigen Informationen für eine lokale Proxy-Konfiguration an. Zusätzlich kann ein (einfacher!) Passwortschutz konfiguriert werden.

Nach der Einrichtung des SOCKS-Proxy wird das SlaVaComp-Beispiel unter seiner ursprünglichen IP unter http://10.0.2.100 verfügbar sein und die Ports 80 und 8080 freigeben. Beispiel CiTAR SlaVaCom Landingpage (SOCKS mode) (hdl:11270/62fd18be-44ec-4b1d-94b4-887fa139142815)

3. Local-mode In manchen Situationen ist eine vollständig private Verbindung erforderlich. Zusätzlich soll es möglich sein, eine archivierte Maschine in aktuelle (virtuelle) Forschungsumgebungen einzubinden. Hierfür bietet CiTAR einen lokalen Verbindungsmodus an. Der Benutzer muss ein kleines Programm (verfügbar für Linux, MacOS und Windows) installieren, das einen CiTAR-Netzwerk URL-Handler registriert.

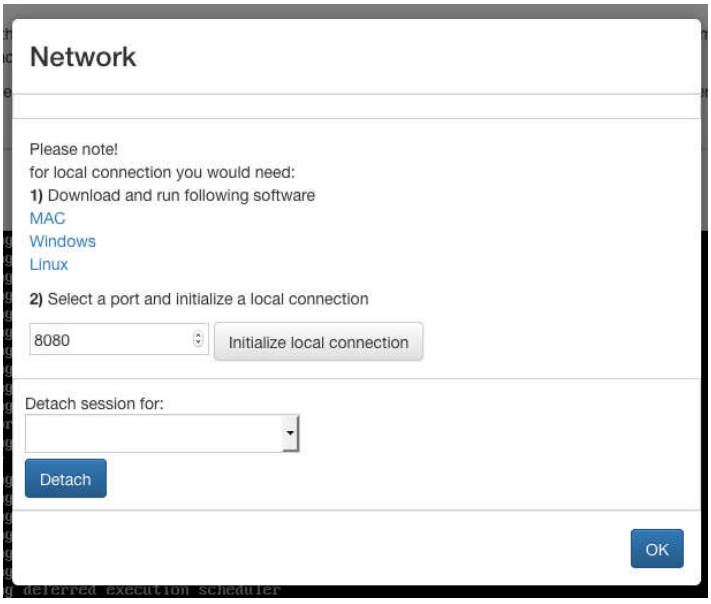

Abb. 3: Einrichten einer privaten Netzwerkverbindung.

Der Benutzer kann dann mit den archivierten Netzwerkumgebungen interagieren, z.B. eine aktuelle Excel-Tabelle mit einem archivierten Datenbankserver verbinden oder im Falle von SlaVaComp mit der REST-Schnittstelle über lokale Programme oder Skripte interagieren. CiTAR SlaVaComp Landigpage (local mode) (hdl: 11270/62ddddddd-44ec-4b1d-94b4-887fa139142815)

Erhaltung von Containern

Die Containertechnologie (auch Betriebssystemvirtualisierung genannt) wurde als leichtgewichtige Alternative zu virtuellen Maschinen entwickelt. Docker und (in den wissenschaftlichen Communities insbesondere) Singularity[7], wurden recht schnell von Forschenden aufgegriffen, da Container in der Lage sind, eine Softwareumgebung, z.B. eine komplexe Softwareinstallation, in eigenständige portable Einheiten zu kapseln. Um wissenschaftliche Methoden in Containern langfristig zugänglich, nutzbar und zitierbar zu machen, muss deren Langlebigkeit als neue digitale Objektklasse gewährleistet sein.

Aufgrund der strengen Isolation (Portabilitätsanforderung) müssen Container eigenständig sein, d.h. alle Softwareabhängigkeiten (Bibliotheken, Anwendungen etc.) müssen innerhalb des Containers aufgelöst sein. Somit ist die Hauptkomponente eines Containers ein in sich abgeschlossenes Dateisystem mit installierten und konfigurierten Softwarekomponenten.

Die einzige verbleibende externe Softwareabhängigkeit ist das zugrunde liegende Betriebssystem, d.h. das Application Binary Interface (ABI) des Betriebssystems. Die zweite Containerkomponente ist die Laufzeitkonfiguration, die Dateisystem-Mappings, z.B. gemeinsame Ordner zwischen Container und Hostsystem, sowie die Definition eines Einstiegspunktes innerhalb des Containers (z.B. ein Skript oder Programm) definiert.

[7] https://www.sylabs.io/docs/.

Struktur eines Containerobjekts

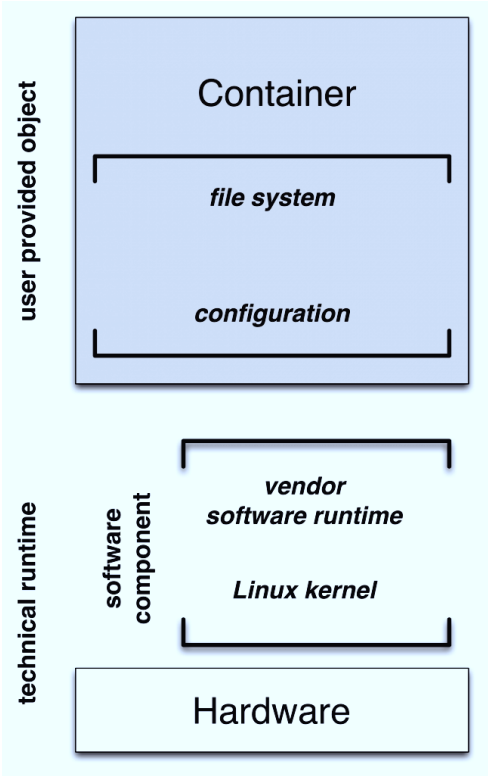

Abb. 4: Container. Eine neue Klasse von digitalen Objekten.

Die technische Laufzeitumgebung eines Containers setzt sich also aus zwei Komponenten zusammen: einer Hardwarekomponente (dem Computer) und einer Softwarekomponente. Im Allgemeinen stellt die Softwarekomponente eine Linux-Installation mit einer installierten und konfigurierten (herstellerspezifischen) Container-Laufzeitumgebung dar. Die Erhaltung von Containern und deren technischen Laufzeit kann somit entkoppelt werden, einerseits durch das Erstellen und Erhalten eines (generischen) Container-Laufzeit-Setups und andererseits dem Ersetzen veralteter Computer durch virtuelle oder emulierte Hardware. Für letzteres wird ebenfalls EaaS als grundlegende Technologie eingesetzt.

Obwohl Container auf den gleichen technischen Grundlagen aufbauen, verwenden die verschiedenen Containerimplementierungen wie Docker oder Singularity unterschiedliche Containerformate und Konfigurationsformen.

Diese Formatunterschiede erfordern in der Regel herstellerspezifische technische Laufzeitumgebungen. Im Idealfall sollte jedoch eine kleine Anzahl von Software-Laufzeitumgebungen ausreichen, um eine große Anzahl von Containern zu betreiben und damit den zukünftigen Verwaltungs- und Wartungsaufwand reduzieren. Da man davon ausgehen kann, dass veröffentliche und insbesondere archivierte Container final sind, werden die Containerformate verallgemeinert, indem herstellerspezifische Komponenten in eine gemeinsame, (teilweise) standardisierte Struktur migriert wurden, so dass aus Erhaltungssicht alle archivierten Containerobjekte dieselben technischen Eigenschaften, d.h. technischen Abhängigkeiten, aufweisen. Darüber hinaus kann man davon ausgehen, dass die Betriebssystemschnittstelle selten inkompatibel wird, so dass nur wenige Variationen solcher Laufzeiten überhaupt notwendig sind.

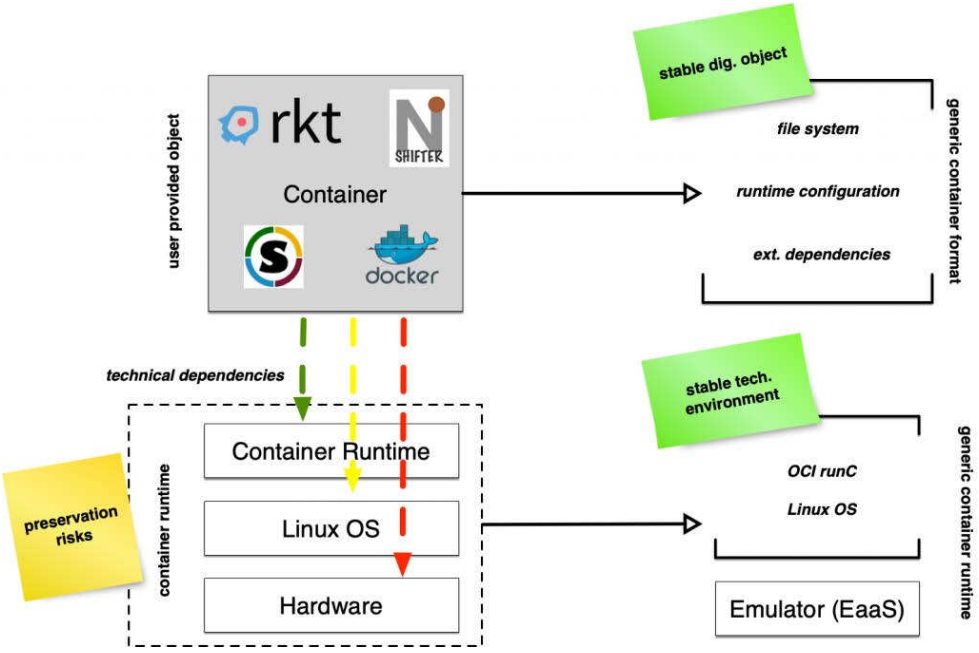

Abb. 5: Aufbau eines Containers.

Publikation zitierfähiger Container

Um zu veranschaulichen, wie Container in CiTAR publiziert und nachgenutzt werden können, soll hier ein Tutorial-Workflow, der von den Entwicklern von OpenMS für die

Workflow-Engine KNIME (Konstanz Information Miner) zur Verfügung gestellt wird, beschrieben werden.

OpenMS ist eine Open-Source-Software zur Verarbeitung von Massenspektrometrie-Daten. Mit KNIME kann man Workflows erstellen, indem man *Knoten* hinzufügt und verbindet. Knoten bieten konfigurierbare Funktionen zur Verarbeitung von Daten im Workflow. Der Benutzer kann zusätzliche Knoten installieren, um KNIME um Funktionen zu erweitern. Die OpenMS-Entwickler haben KNIME-Knoten erstellt, die in dem hier vorgestellten Beispiel verwendet werden.

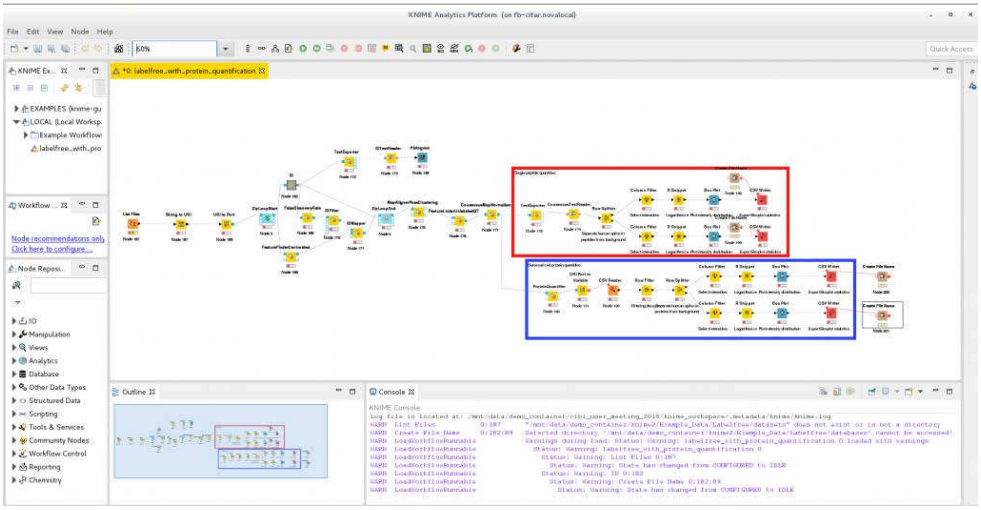

Abb. 6: KNIME-Workflow innerhalb eines Singularity-Containers.

Der in diesem Beispiel erstellte Workflow wird als Singularity Container exportiert und kann anschließend in CiTAR durch den Forschenden selbst importiert werden (vgl. Abb. 6). Hierfür muss der Container zunächst technisch beschrieben werden, insbesondere, wie der enthaltene Prozess gestartet werden soll, wo die Eingabedaten erwartet werden und wo die Ausgabedaten hinterlegt werden. Zusätzlich soll die Funktionsweise des Containers für spätere Nutzer beschrieben werden. Diese Beschreibung bildet dann die Grundlage für die von CiTAR erzeugte Landingpage.

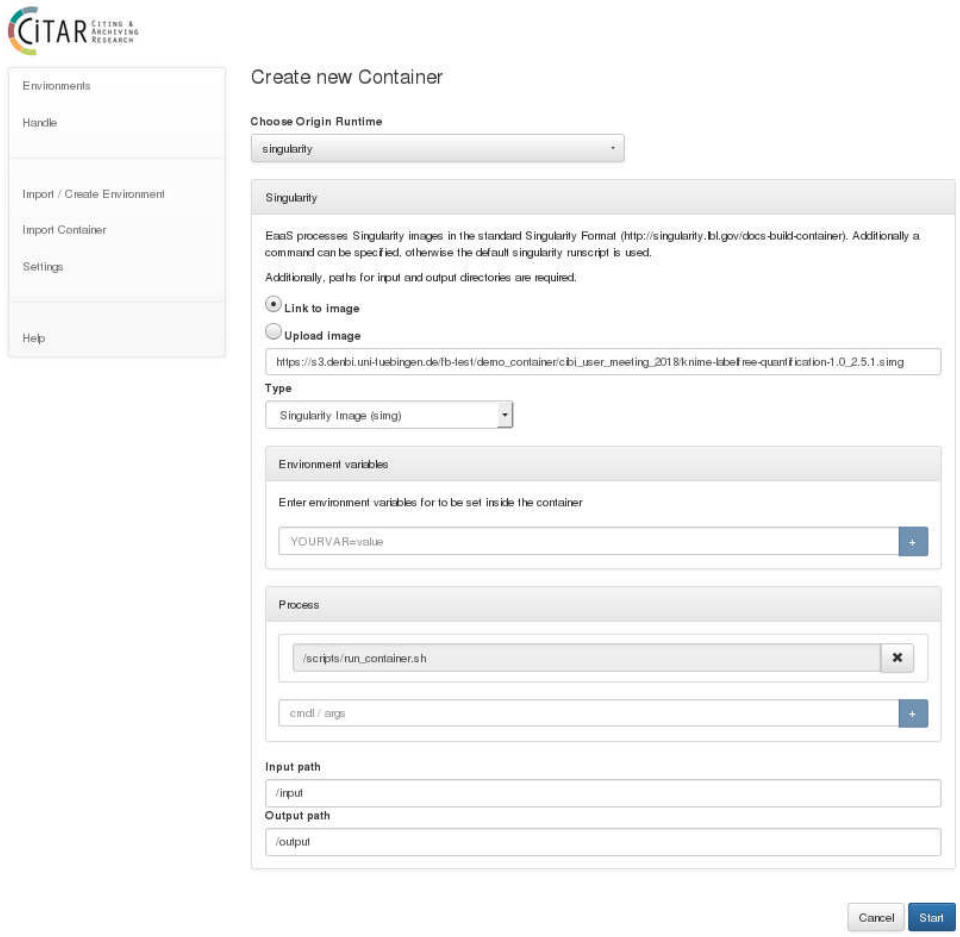

Abb. 7: Erstellung eines neuen Containers und Übermittlung an CiTAR.

Nachnutzung der publizierten Container

Der importierte Container beinhaltet einen Workflow der OMSSA, ein Software Framework zur Massenspektrometrie, die auf einer Sequenzdatenbank basiert. Die von CiTAR erstellte Landingpage (http://hdl.handle.net/11270/188b6194-ea68-4e59-88c2-61d652941e29) beschreibt dessen Funktionsweise und beinhaltet unter anderem einen Link zu einem Beispieldatensatz und eine Beispielausgabe des Workflows. Die notwendigen Datensätze werden nicht von CiTAR selbst verwaltet und sollen als Parameter beim Start des Container als Input-Parameter angegeben werden.

Abb. 8: Reproduktion von Ergebnissen.

Ausblick

Die Erhaltung von softwarebasierten wissenschaftlichen Methoden stellt eine wesentliche Grundlage für reproduzierbare Forschung dar. Im Projekt CiTAR wurden dazu erste Lösungsansätze entwickelt, um Container und andere virtuelle Forschungsumgebungen (u.a. virtuelle Maschinen) langfristig zugänglich zu machen und insbesondere in aktuelle Prozesse zu integrieren. Nachdem die Software fertiggestellt und sie anhand ausgewählter Beispiele erprobt wurde, wird derzeit ein öffentliches Portal entwickelt, das im Laufe des Jahres Interessierten zur Verfügung stehen wird.

Literatur

Garijo, D. / Kinnings, S. / Xie, L. / Xie, L. / Zhang, Y., Bourne, P. E. / Gil, Y. (2013): *Quantifying reproducibility in computational biology: The case of the tuberculosis drugome.* PLoS ONE, 8(11), 1–11. https://doi.org/10.1371/journal.pone.0080278

Brase, J. (2009): “DataCite -- A global registration agency for research data.” In: *Cooperation and Promotion of Information Resources in Science and Technology, 2009. COINFO'09. Fourth International Conference on* IEEE (pp. 257-261).

Meng, Haiyan/Kommineni, Rupa/Pham, Quan/Gardner, Robert/Malik, Tanu / Thain, Douglas: "An invariant framework for conducting reproducible computational science." In: *Journal of Computational Science* 9 (2015): 137-142.

Boettiger, Carl (2015): "An introduction to Docker for reproducible research." In: *ACM SIGOPS Operating Systems Review* 49, 1 (2015): 71-79.

Rechert, Klaus/Valizada, I. /Suchodoletz, D. von/Latocha, J.: "bwFLA – A functional approach to digital preservation." In: *PIK-Praxis der Informationsverarbeitung und Kommunikation* 35, 4 (2012): 259.

Rechert, Klaus/Falcao, P. / Emson, T.. “Towards a Risk Model for Emulation-based Preservation Strategies: A Case Study from the Software-based Art Domain.” In: *Digital Preservation (iPRES) 2016 13th International Conference on*, 139 - 148 (2016).

Rechert, Klaus/Liebetraut, T. / Kombrink S. / Wehrle, D./ Mocken, S. / Rohland, M.: “Preserving Container”. In Kratzke, J. und Heuveline, V. (Hrsg.)*: E-Science-Tage 2017: Forschungsdaten managen,* Heidelberg: heiBOOKS 2017.

The CiTAR Team

Rafael Gieschke, Susanne Mocken, Klaus Rechert, Oleg Stobbe, Oleg Zharkov (University Freiburg), Felix Bartusch (University Tübingen), Kyryll Udod (University Ulm)