SAS Container Runtime – mit SAS auf schlankem Fuss unterwegs

30. Plattform Netzwerktreffen
Heidelberg Haarlass
28.09.2023

Hans-Joachim Edert, SAS Global Technology Practice EMEA

Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime (SCR)

Agenda

- SAS Container Runtime (SCR) Introduction
- „A day in the life of a container“ – the SCR Lifecycle
- In-depth SCR
  - Creating the Publishing destination
  - Publishing and validating the container image
  - Running the container image in production
  - REST API calls to access the model in the SCR container

- Live demo

Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime (SCR)

Introduction

Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime (SCR)

Introduction

SAS Container Runtime is a lightweight OCI (Open Container Initiative) compliant container that scores SAS models and decisions. This enables you to run models and decisions on any OCI compliant compute system, including clusters that are based on Docker and Kubernetes. Deployments to cloud or on-premises systems are both supported.

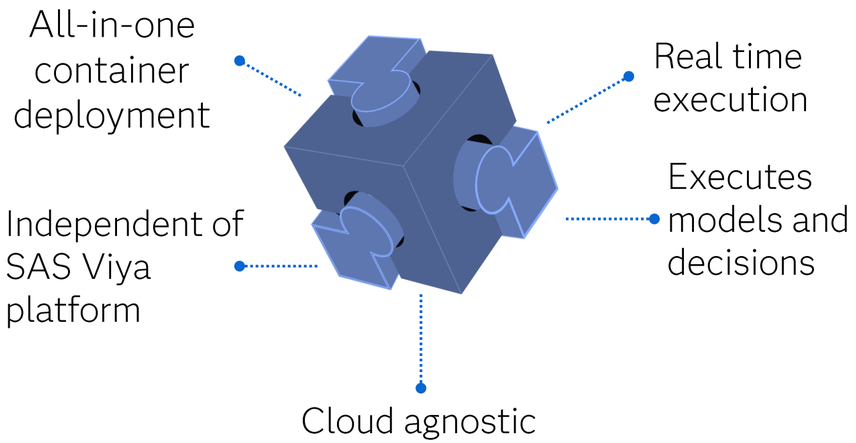
You create a SAS Container Runtime image by publishing models from SAS Model Manager and decisions from SAS Intelligent Decisioning.

- OCI: Industry standard, collection of container runtime, image and distribution specifications
- Can be deployed to all common container runtime environments (Kubernetes, Docker Swarm, AWS ECS, Azure Container Apps, ...)
- Can be handled by all common container tools (docker, podman, ...)

Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime (SCR)

A Liteweight Component for Executing Decisions & Models



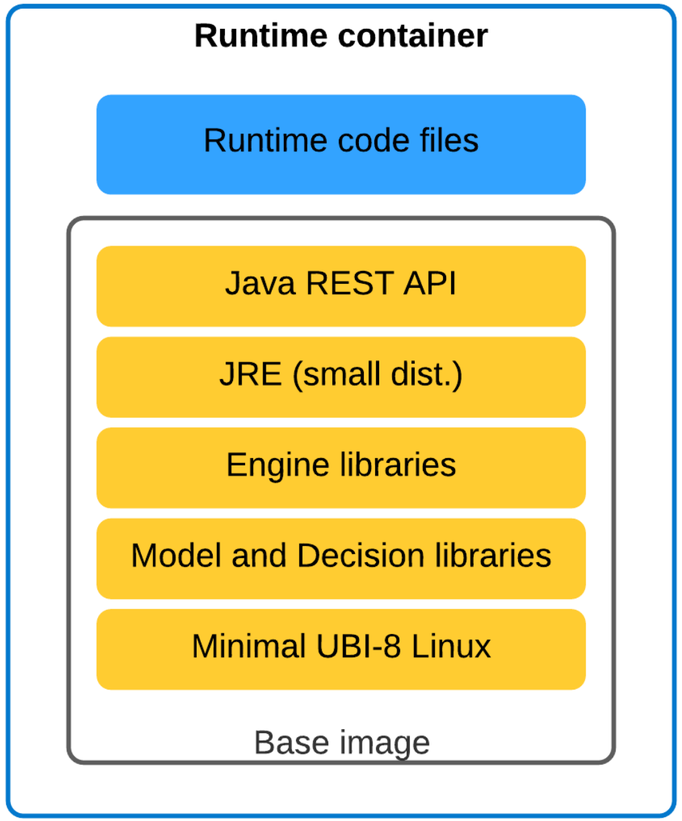
- Provides a new option for deploying decisions/models in production
- Supports high SLAs
- Supports auto scaling, high availability, and disaster recovery
- Requires SAS Model Manager
- No additional fees to customers
- Supports logging and monitoring
- Requires SAS Viya environment for model development and publishing
- Does not require SAS Viya environment during execution of SCR container
- Not an “all-purpose SAS/Base runtime” environment

Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime image details

What's in the SCR image?

- There is one model/decision per Docker container (SCR) image
- Minimal disk and memory footprint
  - Uses small distributions and selected libraries
- Runtime code files are loaded and compiled at container initialization
  - Since there is only one module, start time is reduced (compared to MAS)



Copyright © SAS Institute Inc. All rights reserved.

SAS Container Runtime (SCR)

Capabilities

- Supported Model Score Code Types
  - Analytic store (ASTORE)
  - DATA step
  - DS2 multi-type
  - DS2 package
  - Python
    - A SAS Container Runtime image is created only when a Python model is included in a SAS Intelligent Decisioning decision
    - If you publish a Python model using SAS Model Manager, an open-source image is created which uses a different REST API

- Supported Decision Object Types
  - Assignment rule sets, treatment groups, models, nested decisions, or DS2 code files and custom context files that do not contain SQL queries
  - Filtering rule set nodes
  - Data query files, DS2 code files that contain SQL queries, or custom context files that contain SQL queries
    - Microsoft SQL Server, Oracle, PostgreSQL
  - Python

Copyright © SAS Institute Inc. All rights reserved.

„A day in the life of a container”

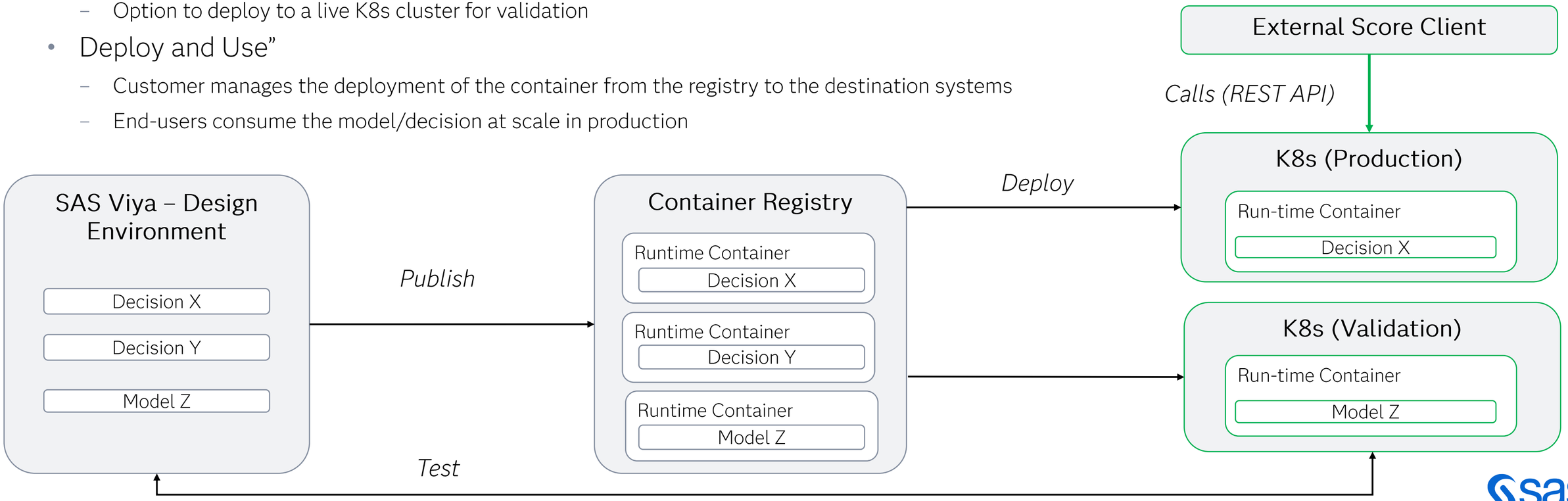
The SCR Lifecycle

Copyright © SAS Institute Inc. All rights reserved.

„A day in the life of SCR“

The SCR Lifecycle

- SAS administrator configures SAS Viya to connect to the customer’s container registry and Kubernetes cluster (k8s)
- “Create and Publish”
  - SAS developer develops models/decisions in SAS Viya
  - SAS developer publishes a model/decision to a container registry using SAS Model Manager
- “Validate”
  - Option to deploy to a live K8s cluster for validation
- Deploy and Use”
  - Customer manages the deployment of the container from the registry to the destination systems
  - End-users consume the model/decision at scale in production

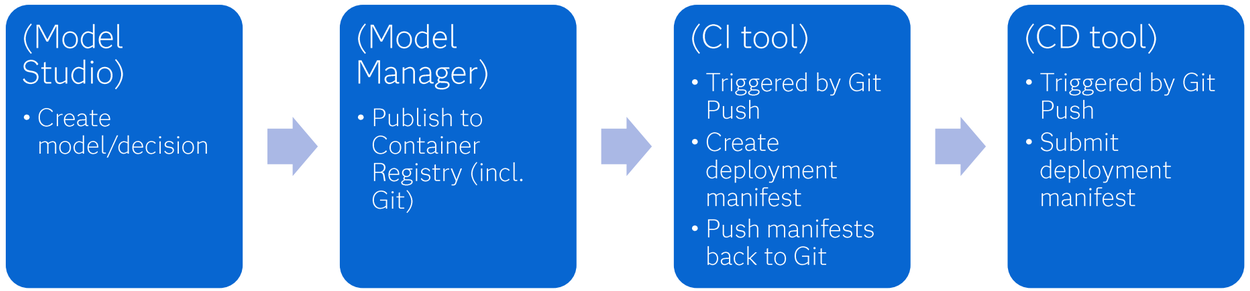


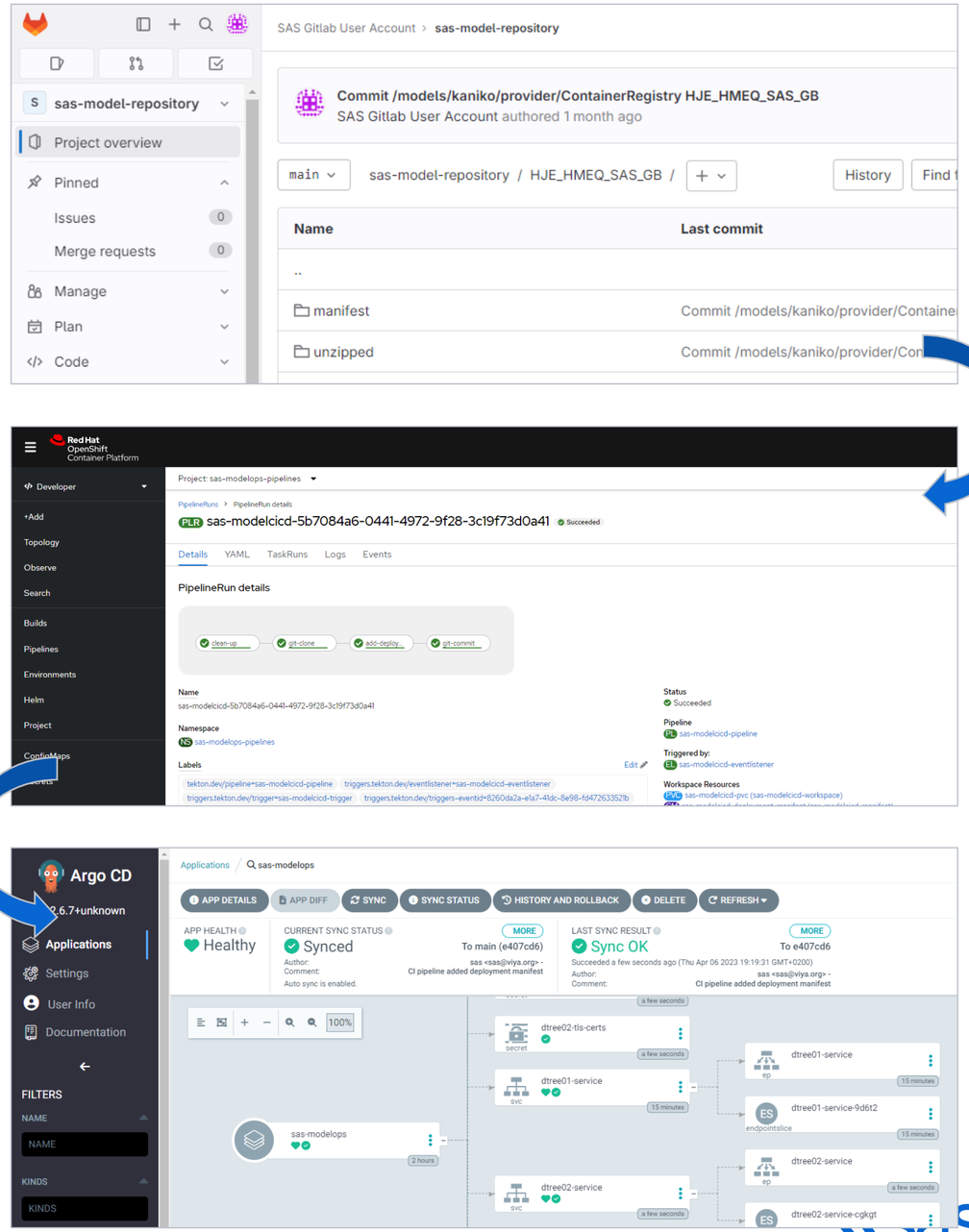
Copyright © SAS Institute Inc. All rights reserved.

„A day in the life of SCR“

The SCR Lifecycle

- Deploying SCR container images to a production environment should be an automated process using pipelines (GitOps)
  - „CI“ – usually a pipeline tool which monitors a Git repository / a container registry
    - Can create the missing YAML manifests for deployments to Kubernetes: Deployment/Pod, Service, Ingress, Secret …
  - „CD“ – usually a tool which synchronizes the target environment with manifest found in a Git repository





Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

Defining the publishing destination

- Use the SAS Viya CLI for this task
- Complex command, does multiple things at the same time. Some are optional, others are not …
  - Defines connection details to Container Registry
  - (Optionally) defines connection details to Git repository
  - (Optionally) defines connection details to Kubernetes cluster for validation

Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

Defining the publishing destination

- Why you should define a connection to a Git repository as well …
  - If defined, SAS Model Manager will push all artefacts used for building the SCR container image to the Git repository (Dockerfile etc.)
  - You can use this information to build the SCR image by yourself, optionally adding your own overlays to the container (e.g. for auditing)
  - You can use the „git push“ event as a trigger for your CI/CD pipelines

- What‘s the purpose of the „model validation“ feature in Model Manager?
  - Model validation is an optional step after publishing to the container registry
  - If configured, Model Manager will temporarily deploy the SCR image to a Kubernetes cluster, send some data to it and monitor the responses
  - After the test has completed, the SCR container will be removed from the cluster
  - It‘s a way to data scientists to check if their model works as expected in the SCR container
  - On Kubernetes clusters with RBAC enabled (e.g. OpenShift), additional privileges are needed to allow Model Manager to deploy the test container

Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

Building the SCR container image

- From a technical perspective, “publishing to SCR” means:
  - Model Manager launches a Kubernetes job which builds the container image
  - The container image is pushed to a container registry
  - If defined, the job also pushes all artefacts to a Git repository
- SCR build job uses the kaniko project
  - Open-source project originally developed by Google
    - https://github.com/GoogleContainerTools/kaniko
  - Runs inside a Kubernetes cluster to build container images from a Dockerfile
  - SAS has packaged it as a microservice and deploys kaniko as part of the SAS Viya software stack
  - Requires elevated permissions on OpenShift. Need to bind the anyuid SCC to the sas-model-publish-kaniko service account

Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

Deploying (Kubernetes)

- On Kubernetes, it is required to create a manifest for deploying the SCR container
- The manifest defines the Kubernetes resources to be created
  - `Deployment`
  - `Service`
  - `Ingress` or `Route` (OpenShift)
  - `Secrets` (image pulls, TLS certificates)
- Review and set the available input and output parameters, e.g. for logging
  - Passed to the container pod as environment variables

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    spec:
      containers:
        - name: my-model-app
          image: myregistry/my-container:v1
          env:
            - name: SAS_SCR_LOG_LEVEL_SCR_IO
              value: "TRACE"
            - name: SAS_SCR_LOG_LEVEL_App.tk.MAS
              value: "DEBUG"
```

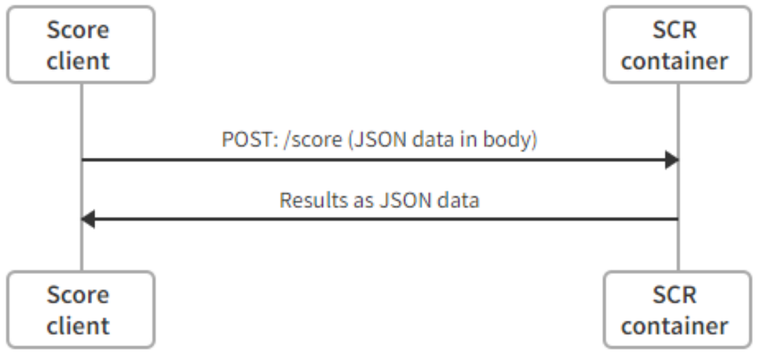
Copyright © SAS Institute Inc. All rights reserved.

In-depth SCR

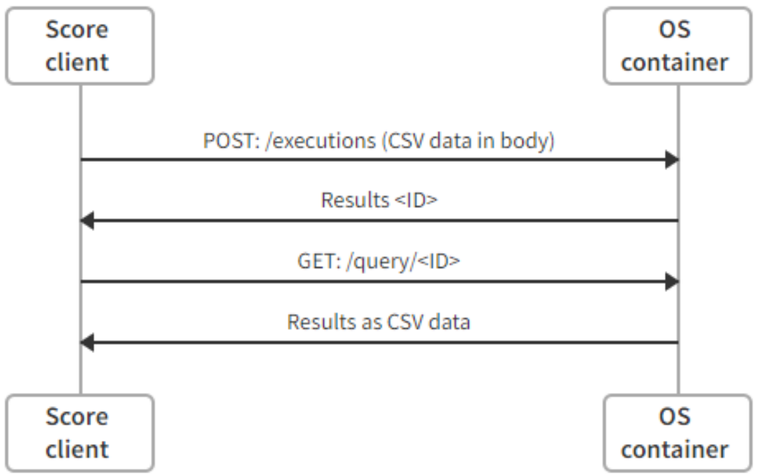
Executing (REST API)

- SCR containers can be used by calling a REST API
- REST API is different for SCR containers and open-source containers
  - Remember that publishing a Python model to a Container registry does not generate a SCR container image

REST API flow for SCR containers



REST API flow for open-source containers



Copyright © SAS Institute Inc. All rights reserved.

Live demo

Copyright © SAS Institute Inc. All rights reserved.

Q & A

SAS Container Runtime (SCR)

Resources

- Using the SAS Container Runtime for publishing SAS models to Kubernetes on the Azure cloud
  - https://communities.sas.com/t5/SAS-Communities-Library/Using-the-SAS-Container-Runtime-for-publishing-SAS-models-to/ta-p/760835
- How to Publish a SAS Model to Azure with SCR: A Start-to-Finish Guide
  - https://communities.sas.com/t5/SAS-Communities-Library/How-to-Publish-a-SAS-Model-to-Azure-with-SCR-A-Start-to-Finish/ta-p/768714
- Automating model delivery with SAS Viya on the Red Hat OpenShift Container Platform
  - https://communities.sas.com/t5/SAS-Communities-Library/Automating-model-delivery-with-SAS-Viya-on-the-Red-Hat-OpenShift/ta-p/877616
- SAS Git repository with SCR resources
  - https://github.com/sassoftware/sas-container-runtime
- SAS SCR Programming and Administration Guide
  - https://go.documentation.sas.com/doc/en-US/mascrtcdc/v_017/mascrtag/n0vobwu1imdrzin1xjuc8v717sw0.htm

Copyright © SAS Institute Inc. All rights reserved.

Thank your for your time

hans-joachim.edert@sas.com

Copyright © SAS Institute Inc. All rights reserved.