1 Inbetriebnahme von WEGA unter Docker

1.1 Was ist Docker

Docker ist eine Serie von Platform as a Service Werkzeugen, die Virtualisierungsmethoden der Betriebssysteme nutzen, um Software in sogenannten Container isoliert betreiben zu können. Diese Container enthalten dabei alle benötigten Abhängigkeiten, wie zum Beispiel Bibliotheken oder Laufzeitumgebungen, um die bereitgestellte Software betreiben zu können. Daher muss außer der Ausführungsumgebung für den Container – im folgenden Fall die Software Docker – nichts weiter installiert werden. Basis für einen Container ist ein sogenanntes Image, das vom Softwareanbieter bereitgestellt wird.
Vorteil der Container ist, dass sie einen genau definierten Softwarestand beinhalten. Dieser Softwarestand kann aufgrund der Isolation nicht von anderen Installationen oder weiterer Software, die auf dem Hostserver installiert ist, beeinflusst werden. Außerdem wird der Inhalt des Containers bei jedem Neustart auf den Auslieferungszustand, wie er im Image war, zurückgesetzt. Somit können ungewollte Änderungen sehr einfach wieder rückgängig gemacht werden. Dies erlaubt auch den einfachen Wechsel zwischen Versionen der eingesetzten Software, da jede Version in Form eines Images bereitgestellt wird. Den Wechsel zwischen den Versionen stellt damit einfach der Wechsel des zugrundeliegenden Images dar. Es ist keine weitere Installation notwendig. Der Wechsel auf ältere Versionen ist somit auch sehr einfach zu realisieren.
Zusätzlich zum fest definierten Stand des Images ist es möglich einzelne Dateien oder komplette Verzeichnisse des Hostservers innerhalb des Containers bereitzustellen. Diese sogenannten Volumes sind dabei vom Zurücksetzen auf den Auslieferungszustand ausgenommen und erlauben damit eine persistente Speicherung von Konfigurationen oder Daten außerhalb des Containers. Damit sind die Dateien innerhalb dieser Volumes einfach anpass- oder austauschbar und liegen im Dateisystem des Hostservers vor.

1.2 WEGA als Docker

Die Bereitstellung von WEGA für Docker wird mittels Orchestrierung von Containern mit dem Werkzeug Docker Compose durchgeführt. Orchestrierung bedeutet hier, dass für den Betrieb von WEGA nicht nur ein Container, sondern mehrere mit verschiedenen Softwarekomponenten gestartet werden, die voneinander abhängig sind. Derzeit stehen die einzelnen Komponenten WEGA inklusive WEGA-3D, der WEGA-AuthService und der MCM Lizenz Manager zur Verfügung. Weitere Komponenten wie zum Beispiel WEGA-EMS-Server sind in Vorbereitung.
Docker Compose wird über eine Steuerdatei konfiguriert, die von uns zur Verfügung gestellt wird. Diese Steuerdatei „docker-compose.yml“ enthält dabei die Definitionen der Volumes, die als persistenter Speicher verwendet werden, verschiedene Umgebungsvariablen für die Container und welche Netzwerk Ports der Container erreichbar sein sollen.

Auszug aus der aktuellen docker-compose.yml:

```
version: '3.5'
services:
  wega:
    image: docker.moss.de/mossde/wega:9.1.2.1
    ports:
      - '8080:8080'
    environment:
      - WEGAAUTH_HOST=auth
      - MCMLIC_HOST=mcmlic.muc.moss.itn  # mcmlic.muc.moss.itn or licserver if the image is used
      - MCMLIC_PORT=27010
      - MCMLIC_USER=wega
      - JAVA_OPTS=-Dhttp.proxyHost=proxy.muc.moss.itn -Dhttp.proxyPort=3128 -Dhttps.proxyHost=proxy.muc.moss.itn -Dhttps.proxyPort=3128
      - USERID=1000

    volumes:
      - ./rba-config:/home/moss/rba-config/
      - ./wega/logs:/home/moss/tomcat/logs/
      - ./wega/base/web.xml:/home/moss/wegabase/WEB-INF/web.xml
      - ./wega/rba/web.xml:/home/moss/wegarba/WEB-INF/web.xml
      - ./wega3d/data:/home/moss/wega-3d-data/
      - ./wega3d/defaultConfig.js:/home/moss/wega3d/defaultConfig.js
      - ./wega3d/config.json:/home/moss/wega3d/config.json
```

Unter dem Schlüsselwort „image“ wird das entsprechend zu verwendende Image für den Container adressiert. Hier handelt es sich um die Version 9.1.2.1 des von uns bereitgestellten Images für WEGA. Die Version kann hierbei durch die Änderung des Namens des Images erreicht werden.
Unter dem Schlüsselwort „port“ werden die Netzwerk Ports des Containers, die von außen erreichbar sein sollen, definiert. Dabei gilt, dass aufgrund der Isolation der Container nur die hier angegeben Ports erreichbar sind. Dabei kann auch eine Änderung des Ports auf zum Beispiel Port 80 an dieser Stelle durchgeführt werden. Die Syntax ist dabei <Port_auf_Host>:<Port_in_Container>.
Unter dem Schlüsselwort „environment“ werden dem Container verschiedene Umgebungsvariablen für den Start mitgegeben. Die wichtigsten hierbei sind der Port und Host des MCM Lizenz Managers mit der Lizenz für WEGA und die zusätzlichen Java-Parameter für den Apache Tomcat. Im obigen Beispiel sind dies die Parameter für die Angabe eines zentralen Proxyservers für das Abrufen von Diensten im Internet.
Unter dem Schlüsselwort „volumes“ werden die Dateien und Verzeichnisse auf dem Hostserver definiert, die dem Container bereitgestellt werden. Hier ist die Syntax <Pfad_auf_Host>:<Pfad_im_Container>. Die Pfade auf dem Hostserver können sowohl relativ ausgehend von der Datei docker-compose.yml oder absolut im Dateisystem angegeben werden. Der Pfad im Container ist durch das Image festgelegt und kann nicht geändert werden. Im obigen Beispiel sind die bekannten Konfigurationsdateien und Pfade von WEGA als Volumes definiert. Zusätzlich ist ein Pfad für die Logdateien definiert, so dass diese ebenfalls außerhalb des Containers gespeichert werden.

1.3 Steuerung

Das Steuerung der Orchestrierung erfolgt dann nach dem Anpassen der Konfiguration mittels:

- docker-compose pull – dieser Befehl lädt die aktuellen Images herunter, die in der Datei benannt sind
- docker-compose up -d – dieser Befehl startet die Container basierend auf den Images und der Konfiguration
- docker-compose down – dies stoppt die definierten Container wieder und setzt den Inhalt auf den Auslieferungszustand der Images zurück.

Alternativ können hierfür auch Oberflächen verwendet werden, wie zum Beispiel die Software Portainer.

1.4 Schlussbemerkung

Mit der Bereitstellung als Image für Docker und durch die Nutzung der Orchestrierung mittels Docker Compose ist es einfacher geworden WEGA in Betrieb zu nehmen. Ebenso werden die Wartungsarbeiten vereinfacht, da neue Versionen von sowohl WEGA als auch der abhängigen Softwarepakete wie Java und Apache Tomcat durch ein neues Image einfach eingespielt werden können.
Die Bereitstellung weiterer Softwarepakete aus den Produktfamilien WEGA und novaFACTORY ist bereits in der Entwicklung.