

DCT Manager - ferramenta *web* para gestão de *templates* Docker Compose

Sérgio Silva¹, Vitor M. R. Cunha¹

¹ Instituto Superior de Engenharia do Porto, Porto, Portugal
{1060423,vrc}@isep.ipp.pt

Resumo O Docker é um *software* que permite a virtualização ao nível da aplicação, executando-as num ambiente isolado chamado *container*. Os *containers* são criados com base numa imagem que especifica os seus conteúdos e que são, regra geral, obtidas a partir de repositórios públicos. O Docker Compose é uma ferramenta que facilita a gestão de ambientes onde são usados vários *containers* em simultâneo, utilizando um único ficheiro que reúne todas as configurações necessárias. No entanto, num cenário complexo que envolva vários ficheiros Docker Compose a sua gestão pode tornar-se uma tarefa demorada e sujeita a erros. Assim, este artigo apresenta uma ferramenta *web* - DCT Manager - que tem como objetivo proporcionar um método simplificado de criação, configuração e manutenção de ficheiros Docker Compose. A abordagem utilizada possibilita a criação de ficheiros *compose* complexos que permitem lançar vários serviços a partir da parametrização de *templates* de serviços individuais. A flexibilidade introduzida pela definição simplificada de *templates* torna a ferramenta adequada a ser utilizada em vários cenários, e por utilizadores com diferentes níveis de conhecimentos da tecnologia subjacente.

Palavras-chave: *Container*, Docker, Docker Compose, Programação *web*.

1 Introdução

A disponibilização flexível e fiável de serviços informáticos em larga escala coloca diversos problemas e desafios aos gestores das infraestruturas de suporte. A virtualização ao nível da aplicação através do *software* Docker [1] tem-se mostrado como uma das formas possíveis de abordar o problema.

O Docker fornece a capacidade de executar uma aplicação num ambiente isolado chamado *container* e entre as atuais implementações de *containers*, o Docker é provavelmente a mais adotada [2,3]. A sua utilização reduz significativamente o tempo entre o desenvolvimento de aplicações/serviços e a sua colocação em produção. Os *containers* são criados com base numa imagem que especifica os seus conteúdos e que são, regra geral, obtidas a partir de repositórios públicos (e.g., [4]).

De forma a agilizar a gestão de ambientes onde são usados vários *containers*, surgiu a ferramenta Docker Compose [5], também disponibilizada pela Docker

Inc. O Docker Compose usa um ficheiro YAML *Ain't Markup Language* (YAML) para configurar os parâmetros necessários para cada *container* a correr, em oposição à execução de comandos isolados `$docker run`.

2 Estado da Arte

Várias ferramentas têm vindo a ser desenvolvidas com o objetivo de facilitar a gestão de *containers* em ambientes de larga escala [6]. A Fig. 1 apresenta um mapa que dá uma visão geral de algumas das ferramentas mais populares do ecossistema Docker. Note-se que apenas são mencionadas ferramentas *open source*, embora existam disponíveis algumas ferramentas comerciais muito úteis (e.g., Docker Datacenter). No mapa as elipses representam as ferramentas, enquanto que as linhas a laranja delimitam áreas funcionais, cuja classificação é apresentada nos retângulos. Uma posição mais próxima do centro da figura, indica que a ferramenta tem um maior número de funcionalidades em relação a outras em zonas mais afastadas. As setas sólidas representam dependências entre ferramentas e as setas tracejadas indicam que a ferramenta é diretamente suportada, tornando-se visível a existência de dependências e interações entre ferramentas [7].

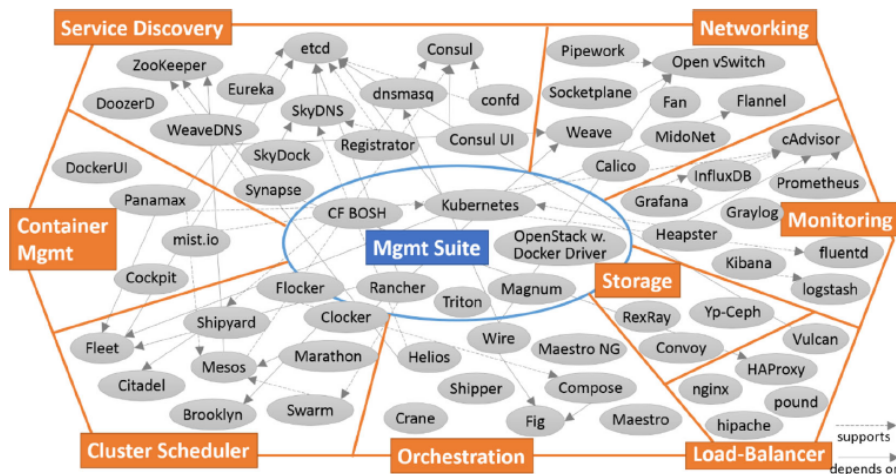


Fig. 1. Ecossistema Docker [7].

As ferramentas da categoria de ‘Orchestration’ de *containers* (e.g., Compose) surgiram com o intuito de, entre outras funções, facilitar a utilização do Docker e são, de forma geral, destinadas a utilizadores com experiência na área e em alguns casos não disponibilizam uma *User Interface* (UI) gráfica. Existem, no entanto, algumas aplicações que permitem a gestão de *containers* (configurados

num ficheiro Docker Compose) através de uma *web* UI. Portainer [8], Docker Compose UI [9] e Seagull [10] são alguns exemplos. Mas a disponibilidade de ferramentas que ofereçam a criação simplificada de ficheiros Docker Compose é limitada, neste conjunto podem mencionar-se: Portainer, que oferece os denominados *app templates* que permitem pré-configurar serviços através da *web* UI da ferramenta. No entanto, cada *template* está limitado a um serviço ou a vários, caso estes se encontrem definidos num *compose file* externo; Composerize [11] que permite converter os argumentos de um comando `$docker run` no correspondente *compose file*; Stacker [12], uma ferramenta que permite lançar serviços configurados num denominado Docker Compose *template* de forma interativa através da linha de comando. Estes *templates* são ficheiros YAML que deverão existir e terem sido criados externamente à ferramenta.

É neste contexto que se identifica a necessidade de ferramentas com UI gráficas e intuitivas, que por este meio proporcionem a utilizadores menos experientes um método simplificado para usufruírem das vantagens da utilização de ficheiros Docker Compose. Este propósito não invalida, no entanto, que uma ferramenta com estas características seja também adequada a utilizadores mais experientes.

3 A ferramenta: DCT Manager

3.1 Objetivos

O objetivo geral da ferramenta Docker Compose Templates (DCT) Manager é o de proporcionar a utilizadores menos experientes na configuração de serviços, uma plataforma acessível que de modo simplificado permita a criação, configuração e manutenção de ficheiros Docker Compose. Assim, a Fig. 2 ilustra onde a ferramenta se enquadra num típico *workflow* de utilização de múltiplos *containers*.

Num cenário de *multi-containers* é vantajoso usar um método que permita a configuração simultânea dos vários serviços a disponibilizar, i.e., o Docker

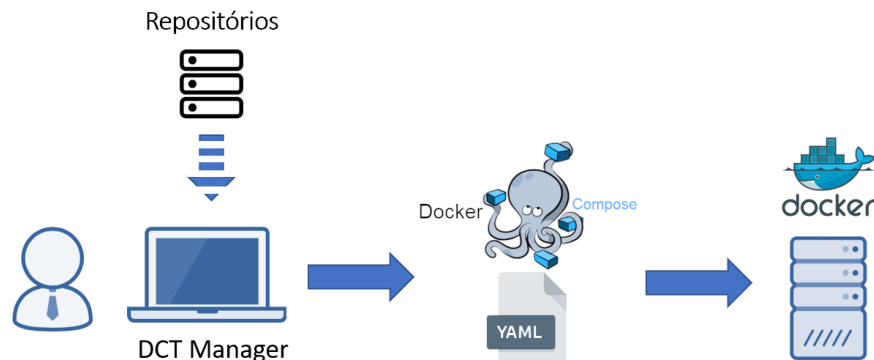


Fig. 2. Enquadramento do funcionamento da ferramenta DCT Manager.

Compose. No entanto, a estrutura deste tipo de ficheiros pode torna-se complexa com o aumento do número de serviços configurados. Adicionalmente, a gestão de vários ficheiros Docker Compose é tendencialmente problemática devido às tarefas associadas à sua manutenção e reutilização. A tarefa de pesquisa de imagens nos repositórios públicos também tem vindo a revelar-se cada vez mais complicada, não só pela necessária interpretação da informação disponibilizada para criação do respetivo Docker Compose, mas também pela crescente oferta e variedade de versões de imagens. De modo a dar resposta ao problema, a ferramenta introduz o conceito de *compose templates*. Assim, o utilizador poderá criar o seu próprio repositório de *templates* de configurações Docker Compose referentes a serviços individuais, e com base na sua adaptação a cada caso de utilização permite construir um *compose file* complexo capaz de lançar vários serviços.

3.2 Desenvolvimento

A ferramenta DCT Manager apresenta uma estrutura modular, onde os módulos são denominados de *templates* (Fig. 3). Um *template* pode ser usado para configurar um ou mais *containers*, e detém toda a informação necessária para gerar um ficheiro Docker Compose. Esta abordagem foi concebida no sentido de facilitar a agregação de serviços que usualmente são usados em conjunto. A ferramenta suporta um número ilimitado de *templates*, sendo também ilimitado o número de *templates* que podem ser selecionados para gerar um *compose file*.

O desenvolvimento da DCT Manager englobou vários passos. Entre os mais relevantes destacam-se a configuração da estrutura, a construção da base de dados, a definição da estrutura de arquivos e o *workflow*. Foram também construídas as necessárias páginas *web* para a interface com o utilizador, tendo para isso sido utilizadas diferentes linguagens, *frameworks* e bibliotecas. Em maior detalhe: *Hypertext Markup Language* (HTML), *Hypertext Preprocessor* (PHP), JavaScript, *Cascading Style Sheets* (CSS), JQuery, Bootstrap e *Structured Query Language* (SQL).

Para a criação da ferramenta foi configurada uma estrutura base que utiliza o *software* Docker e na configuração foram utilizadas as imagens Docker “nginx”, “php”, “mysql” e “phpmyadmin”. Todas estas imagens estão disponíveis no repositório oficial Docker Hub, estando assim assegurada a manutenção das mesmas, tornando a ferramenta globalmente mais estável e segura.

A estrutura da base de dados concebida para suportar as funcionalidades da ferramenta encontra-se dividida em duas tabelas e foi utilizada uma relação do tipo 1 para N [13] para as relacionar. A utilização de duas tabelas teve por objetivo permitir um número ilimitado de serviços (i.e., *containers*) associados a um *template*. Existe uma tabela denominada ‘generalData’ que se destina aos dados que caracterizam o *template* (nome, descrição, *tags*, etc.) e uma tabela ‘composeData’ que contém a informação necessária para a construção do ficheiro Docker Compose (imagens, portos, variáveis de ambiente, etc.).

A funcionalidade de permitir vários *containers* num *template* é útil quando vários serviços são tipicamente usados em conjunto. Por exemplo, quando é ne-

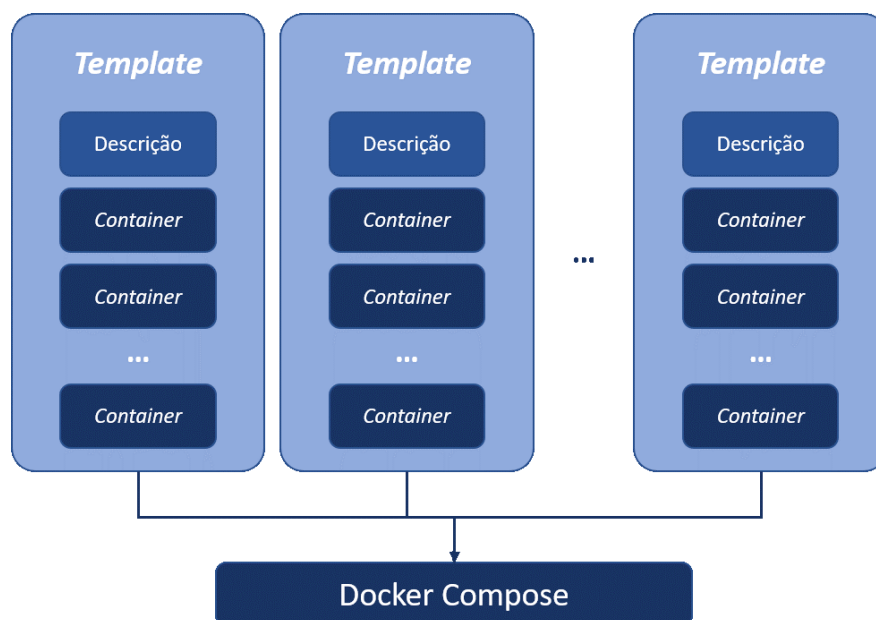


Fig. 3. Estrutura (generalizada) de *templates* e *containers* para gerar um *compose file*.

cessária uma base de dados, usualmente, este *container* é acompanhado de um segundo *container* com uma ferramenta visual para facilitar a sua utilização. A título de exemplo, a DCT Manager oferece um *template* MySQL, que permite lançar em simultâneo os serviços “mysql” (base de dados) e “phpmyadmin” (interface visual de administração).

3.3 Resultados

Depois de lançar o *container* da ferramenta, esta fica disponível num sítio *web* que apresenta um menu inicial com as páginas **Home**, **Insert**, **Edit** e **Delete**. Existe também um menu lateral com opções adicionais. Na página **Home** estão disponíveis todos estes elementos a par com os *templates* configurados na base de dados. A Fig. 4 mostra o *layout* da ferramenta com alguns *templates* exemplificativos, e suas *tags*.

- Página **Home**: nesta página poderá ser feita a seleção dos *templates* que se pretendem agregar para gerar um ficheiro Docker Compose. Para cada *template* é possível ver uma pequena descrição do mesmo através do símbolo no canto superior direito do bloco. Depois de seleccionados os *templates*, surge uma janela onde é possível visualizar e editar todos os parâmetros de configuração definidos nos *templates*.



Fig. 4. DCT Manager: página **Home** com oito *templates* definidos.

- Página **Insert**: na página **Insert** é possível definir novos *templates*. Um *template* pode ser definido simplesmente para lançar um único serviço ou, caso se considere vantajoso, definir vários *containers*. Para além dos parâmetros para cada *container*, deve também ser inserida uma descrição geral do *template* que é apresentada ao utilizador na página **Home** e as *tags* que permitirão encontrar mais facilmente o *template*.
- Página **Edit**: esta página destina-se à alteração da informação dos *templates* existentes na base de dados. Deverá ter-se em consideração que alterar um *template* muda os parâmetros de configuração dos *containers* usados por omissão quando se cria um novo ficheiro Docker Compose.
- Página **Delete**: existe ainda a página **Delete** onde se podem eliminar *templates* definidos na base de dados.
- Filtro de *templates*: por último, a ferramenta disponibiliza a funcionalidade de ser possível filtrar *templates* (menu lateral do lado esquerdo) segundo as *tags* a eles associada, que deverão corresponder à sua principal função. O conjunto de *tags* apresentadas na interface é dinâmico e corresponde ao conjunto de *tags* definidas através das páginas **Insert** ou **Edit**.

4 Conclusões

Pode-se dizer que a DCT Manager permite por um lado, a utilizadores menos experientes instanciar *containers* em Docker apenas através de uma interface

gráfica, e por outro lado, a utilizadores mais experientes criar o seu repositório de serviços, facilitando assim a sua reutilização. A ferramenta pode também ser usada como veículo para disponibilizar *templates* a outros utilizadores ou à comunidade em geral.

A utilidade da ferramenta pode ainda ser evidenciada se se considerar, por exemplo, a configuração de serviços (i) num servidor comum, (ii) num servidor usando o Docker com a utilização tradicional do Docker Compose e (iii) com a utilização da DCT Manager. Considerando a instalação num servidor comum, para além da instalação e configuração do próprio sistema operativo, são necessários vários procedimentos como: a instalação e a configuração das várias aplicações que vão fornecer os serviços, resolução de dependências ou atualizações das aplicações. Para (ii), a configuração dos serviços a disponibilizar teria que ser definida através do formalismo YAML num ficheiro Docker Compose (e.g., `docker-compose.yml`) para depois poder ser usado no comando `$docker-compose up` a executar no terminal do servidor.

No caso (iii), e considerando que a DCT Manager tem os *templates* necessários na base de dados, a sua utilização torna todo o processo de instalação muito mais simples e intuitivo para um utilizador menos experiente. Neste caso, seria apenas necessário selecionar os *templates* desejados, confirmar a informação que lhe é apresentada, e por fim executar o comando `$docker-compose up` no terminal. Neste exemplo considera-se que o utilizador não necessita de alterar nenhum dos parâmetros *default* dos *containers* definidos no(s) *template(s)*.

A configuração inicial dos *templates* é naturalmente a parte mais trabalhosa e que carece de maiores conhecimentos. No entanto, esse trabalho também é sempre necessário, e por vezes superior, através de qualquer um dos cenários anteriormente comparados. A vantagem da ferramenta aqui apresentada é que depois de inseridos os *templates* a sua utilização é simples e facilmente distribuível.

Referências

1. Docker: Site oficial. <https://www.docker.com>
2. Combe, T., Martin, A., Di Pietro, R.: To docker or not to docker: A security perspective. *IEEE Cloud Computing* **3**(5), 54–62 (2016)
3. Netto, H.V., Lung, L.C., Correia, M., Luiz, A.F., de Souza, L.M.S.: State machine replication in containers managed by kubernetes. *Journal of Systems Architecture* **73**, 53–59 (2017)
4. Docker hub. <https://hub.docker.com>
5. Docker compose. <https://docs.docker.com/compose/>
6. Awesome docker: a curated list of docker resources and projects. <https://project-awesome.org/veggemonk/awesome-docker>
7. Peinl, R., Holzschuher, F., Pfitzer, F.: Docker cluster management for the cloud-survey results and own solution. *Journal of Grid Computing* **14**(2), 265–282 (2016)
8. Portainer. <https://www.portainer.io/documentation/>
9. Docker compose ui. <https://github.com/francescou/docker-compose-ui>
10. Seagull. <https://github.com/tobegit3hub/seagull>
11. Composerize. <https://www.composerize.com/>

12. Stacker. <https://github.com/stacker/stacker-cli>
13. Kleppmann, M.: Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems. O'Reilly Media, Inc. (2017)